Agilent Technologies E1460A 64-Channel Relay Multiplexer Module User's Manual

Agilent Technologies

Manual Part Number: E1460-90006
Printed September 2012
Printed in Malaysia E0912

NOTICE: In August 2014, Agilent Technologies' former Test and Measurement business became Keysight Technologies. This document is provided as a courtesy but is no longer kept current and thus will contain historical references to Agilent. For more information, go to **www.keysight.com.**

KEYSIGHT TECHNOLOGIES

Contents

E1460A Relay Multiplexer User's Manual

Front Matter **7**
- Agilent Technologies Warranty Statement 7
- Safety Symbols 8
- Warnings 8

Chapter 1- Getting Started **11**
- Using This Chapter 11
- Multiplexer Description 11
  - Multiplexer Components 11
  - Channel Relay Switches 12
  - Control Relays 13
  - Basic Operating Modes 14
- Configuring the Multiplexer 15
  - Warnings and Cautions 15
  - Setting the Logical Address Switch 16
  - Setting the Status Register Switch 17
  - Setting the Interrupt Priority 17
  - Configuring the Switch Card Wire Jumpers 18
  - Installing the Multiplexer in a Mainframe 21
  - Connecting the Analog Bus 22
- Configuring Terminal Modules 23
  - Standard Terminal Module Description 23
  - Terminal Module Option A3E Description 23
  - Connecting User Inputs 25
  - Wiring Terminal Modules 26
  - Attaching Terminal Modules to the Multiplexer 28
- Programming the Multiplexer 29
  - Checking SCPI Drivers 29
  - Multiplexer Addressing 30
  - Initial Operation 34

Chapter 2 - Using the Relay Multiplexer **35**
- Using This Chapter 35
- Multiplexer Commands/States 35
- Switching Channels 37
  - Switching Channels Comments 37
  - Switching Channels Examples 38
- Scanning Channels 43
  - Scanning Channels Comments 43
  - Scanning Channels Examples 44
- Miscellaneous Multiplexer Functions 51
  - Using the Scan Complete Bit 51
  - Using the Analog Bus 52
  - Saving and Recalling States 56
  - Detecting Error Conditions 56
  - Synchronizing the Multiplexer 58

Chapter 3 - Relay Multiplexer Command Reference 59
About This Chapter 59
Command Types 59
SCPI Commands Reference 61
ABORt 62
ARM 63
ARM:COUNt 63
ARM:COUNt? 64
INITiate 65
INITiate:CONTinous 65
INITiate:CONTinuous? 66
INITiate[:IMMediate] 66
OUTPut 68
OUTPut:ECLTrgn[:STATe] 68
OUTPut:ECLTrgn[:STATe]? 69
OUTPut[:EXTernal][:STATe] 69
OUTPut[:EXTernal][:STATe]? 70
OUTPut:TTLTrgn[:STATe] 70
OUTPut:TTLTrgn[:STATe]? 71
[ROUTe:] 72
[ROUTe:]CLOSe 72
[ROUTe:]CLOSe? 74
[ROUTe:]FUNCtion 75
[ROUTe:]FUNCtion? 76
[ROUTe:]OPEN 77
[ROUTe:]OPEN? 79
[ROUTe:]SCAN 79
[ROUTe:]SCAN:MODE 80
[ROUTe:]SCAN:MODE? 82
[ROUTe:]SCAN:PORT 82
[ROUTe:]SCAN:PORT? 83
STATus 84
STATus:OPERation:CONDition? 86
STATus:OPERation:ENABle 86
STATus:OPERation:ENABle? 86
STATus:OPERation[:EVENt]? 87
STATus:PRESet 87
SYSTem 88
SYSTem:CDEScription? 88
SYSTem:CPON 89
SYSTem:CTYPe? 89
SYSTem:ERRor? 90
TRIGger 91
TRIGger[:IMMediate] 91
TRIGger:SLOPe 92
TRIGger:SLOPe? 92
TRIGger:SOURce 92
TRIGger:SOURce? 94
IEEE 488.2 Common Commands Reference 95
SCPI Commands Quick Reference 96

Appendix A - Relay Multiplexer Specifications 97

Appendix B - Register-Based Programming 99
About This Appendix 99
Register Addressing 99
The Base Address 99
Register Descriptions 102
The WRITE Registers 102
The READ Registers 102
Status/Control Register 103
ID and Device Type Registers 104
Relay Control Registers 104
Programming Examples 107

Appendix C - Relay Multiplexer Error Messages 115

Index 117

Notes:

AGILENT TECHNOLOGIES WARRANTY STATEMENT

AGILENT PRODUCT: E1460A 64-Channel Relay Multiplexer Module **DURATION OF WARRANTY:** 3 years

1. Agilent Technologies warrants Agilent hardware, accessories and supplies against defects in materials and workmanship for the period specified above. If Agilent receives notice of such defects during the warranty period, Agilent will, at its option, either repair or replace products which prove to be defective. Replacement products may be either new or like-new.

2. Agilent warrants that Agilent software will not fail to execute its programming instructions, for the period specified above, due to defects in material and workmanship when properly installed and used. If Agilent receives notice of such defects during the warranty period, Agilent will replace software media which does not execute its programming instructions due to such defects.

3. Agilent does not warrant that the operation of Agilent products will be interrupted or error free. If Agilent is unable, within a reasonable time, to repair or replace any product to a condition as warranted, customer will be entitled to a refund of the purchase price upon prompt return of the product.

4. Agilent products may contain remanufactured parts equivalent to new in performance or may have been subject to incidental use.

5. The warranty period begins on the date of delivery or on the date of installation if installed by Agilent. If customer schedules or delays Agilent installation more than 30 days after delivery, warranty begins on the 31st day from delivery.

6. Warranty does not apply to defects resulting from (a) improper or inadequate maintenance or calibration, (b) software, interfacing, parts or supplies not supplied by Agilent, (c) unauthorized modification or misuse, (d) operation outside of the published environmental specifications for the product, or (e) improper site preparation or maintenance.

7. TO THE EXTENT ALLOWED BY LOCAL LAW, THE ABOVE WARRANTIES ARE EXCLUSIVE AND NO OTHER WARRANTY OR CONDITION, WHETHER WRITTEN OR ORAL, IS EXPRESSED OR IMPLIED AND AGILENT SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTY OR CONDITIONS OF MERCHANTABILITY, SATISFACTORY QUALITY, AND FITNESS FOR A PARTICULAR PURPOSE.

8. Agilent will be liable for damage to tangible property per incident up to the greater of $300,000 or the actual amount paid for the product that is the subject of the claim, and for damages for bodily injury or death, to the extent that all such damages are determined by a court of competent jurisdiction to have been directly caused by a defective Agilent product.

9. TO THE EXTENT ALLOWED BY LOCAL LAW, THE REMEDIES IN THIS WARRANTY STATEMENT ARE CUSTOMER'S SOLE AND EXLUSIVE REMEDIES. EXCEPT AS INDICATED ABOVE, IN NO EVENT WILL AGILENT OR ITS SUPPLIERS BE LIABLE FOR LOSS OF DATA OR FOR DIRECT, SPECIAL, INCIDENTAL, CONSEQUENTIAL (INCLUDING LOST PROFIT OR DATA), OR OTHER DAMAGE, WHETHER BASED IN CONTRACT, TORT, OR OTHERWISE.

FOR CONSUMER TRANSACTIONS IN AUSTRALIA AND NEW ZEALAND: THE WARRANTY TERMS CONTAINED IN THIS STATEMENT, EXCEPT TO THE EXTENT LAWFULLY PERMITTED, DO NOT EXCLUDE, RESTRICT OR MODIFY AND ARE IN ADDITION TO THE MANDATORY STATUTORY RIGHTS APPLICABLE TO THE SALE OF THIS PRODUCT TO YOU.

U.S. Government Restricted Rights

The Software and Documentation have been developed entirely at private expense. They are delivered and licensed as "commercial computer software" as defined in DFARS 252.227- 7013 (Oct 1988), DFARS 252.211-7015 (May 1991) or DFARS 252.227-7014 (Jun 1995), as a "commercial item" as defined in FAR 2.101(a), or as "Restricted computer software" as defined in FAR 52.227-19 (Jun 1987)(or any equivalent agency regulation or contract clause), whichever is applicable. You have only those rights provided for such Software and Documentation by the applicable FAR or DFARS clause or the Agilent standard software agreement for the product involved.

Agilent Technologies

E1460A 64-Channel Relay Multiplexer Module User's Manual
Edition 6
Copyright © 1990, 1992-1995, 2001 Agilent Technologies, Inc. All rights reserved.

Documentation History

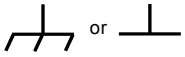
All Editions and Updates of this manual and their creation date are listed below. The first Edition of the manual is Edition 1. The Edition number increments by 1 whenever the manual is revised. Updates, which are issued between Editions, contain replacement pages to correct or add additional information to the current Edition of the manual. Whenever a new Edition is created, it will contain all of the Update information for the previous Edition. Each new Edition or Update also includes a revised copy of this documentation history page.

Edition 1 . . . . . January, 1990
Edition 2 . . . . . July, 1992
Edition 3 . . . . . August, 1993
Edition 4 . . . . . October, 1994
Edition 5 . . . . . November, 1995
Edition 6 . . . . . January, 2001
Edition 6 Rev. 1 . . . . . September 2012

Safety Symbols

Instruction manual symbol affixed to product. Indicates that the user must refer to the manual for specific WARNING or CAUTION information to avoid personal injury or damage to the product.

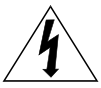
Indicates the field wiring terminal that must be connected to earth ground before operating the equipment — protects against electrical shock in case of fault.



Frame or chassis ground terminal—typically connects to the equipment's metal frame.

Alternating current (AC)

Direct current (DC).



Warning. Risk of electrical shock.

WARNING Calls attention to a procedure, practice, or condition that could cause bodily injury or death.

CAUTION Calls attention to a procedure, practice, or condition that could possibly cause damage to equipment or permanent loss of data.

WARNINGS

The following general safety precautions must be observed during all phases of operation, service, and repair of this product. Failure to comply with these precautions or with specific warnings elsewhere in this manual violates safety standards of design, manufacture, and intended use of the product. Agilent Technologies assumes no liability for the customer's failure to comply with these requirements.

Ground the equipment: For Safety Class 1 equipment (equipment having a protective earth terminal), an uninterruptible safety earth ground must be provided from the mains power source to the product input wiring terminals or supplied power cable.

DO NOT operate the product in an explosive atmosphere or in the presence of flammable gases or fumes.

For continued protection against fire, replace the line fuse(s) only with fuse(s) of the same voltage and current rating and type. DO NOT use repaired fuses or short-circuited fuse holders.

Keep away from live circuits: Operating personnel must not remove equipment covers or shields. Procedures involving the removal of covers or shields are for use by service-trained personnel only. Under certain conditions, dangerous voltages may exist even with the equipment switched off. To avoid dangerous electrical shock, DO NOT perform procedures involving cover or shield removal unless you are qualified to do so.

DO NOT operate damaged equipment: Whenever it is possible that the safety protection features built into this product have been impaired, either through physical damage, excessive moisture, or any other reason, REMOVE POWER and do not use the product until safe operation can be verified by service-trained personnel. If necessary, return the product to Agilent for service and repair to ensure that safety features are maintained.

DO NOT service or adjust alone: Do not attempt internal service or adjustment unless another person, capable of rendering first aid and resuscitation, is present.

DO NOT substitute parts or modify equipment: Because of the danger of introducing additional hazards, do not install substitute parts or perform any unauthorized modification to the product. Return the product to Agilent for service and repair to ensure that safety features are maintained.

Declaration of Conformity

Declarations of Conformity for this product and for other Agilent products may be downloaded from the Internet. There are two methods to obtain the Declaration of Conformity:

- Go to http://regulations.corporate.agilent.com/DoC/search.htm . You can then search by product number to find the latest Declaration of Conformity.
- Alternately, you can go to the product web page (www.agilent.com/find/E1460A), click on the Document Library tab then scroll down until you find the Declaration of Conformity link.

Notes:

Chapter 1
Getting Started

Using This Chapter

This chapter describes the E1460A 64-Channel Relay Multiplexer module, shows how to connect external wiring, and shows how to get started programming the module using Standard Commands for Programmable Instruments (SCPI). This chapter includes:

- Multiplexer Description 11
- Configuring the Multiplexer 15
- Configuring Terminal Modules 23
- Programming the Multiplexer 29

Multiplexer Description

The E1460A is a VXIbus C-Size register-based product that provides switching (multiplexing) of up to 64 two-wire channels. Switching consists of connecting a channel's HI and/or LO line to COM in that bank. The multiplexer can operate in a C-Size VXIbus mainframe using a command module (such as an E1406A Command Module).

Multiplexer Components

The E1460A 64-Channel Relay Multiplexer module consists of a relay switch card and a standard screw-type terminal module. The E1460A is also available with Option A3E that provides a crimp-and-insert terminal housing and connectors. Various configurations can be set by programming (closing) certain switch card relays, and/or selection of wire jumpers on the relay switch card and terminal module.

The E1460A is used when high switch densities such as wire harness/cable testing, semiconductor testing, and/or printed-circuit board testing is required. Although it is primarily a dual 32-channel two-wire multiplexer, the module can be configured to perform one-wire, two-wire, three-wire, and four-wire functions.

By using switch card wire jumpers, the banks can be changed from 1x32 to groups of 1x16 or 1x8. See "Configuring the Switch Card Wire Jumpers" for more information.

For a SCPI environment, one or more multiplexer modules can be defined as a *switchbox* or as a *scanning multimeter*. For a switchbox configuration, all multiplexer channels within the instrument can be addressed using a single interface address. For a scanning multimeter configuration, both the multimeter and all multiplexer modules within the instrument can be addressed using a single interface address.

Channel Relay Switches

The channel relay switches are separated into eight banks. Each bank has eight switchable channels and a COM channel. Each channel has a separate HI (H) and LO (L) line. See Figure 1-1 for a block diagram.

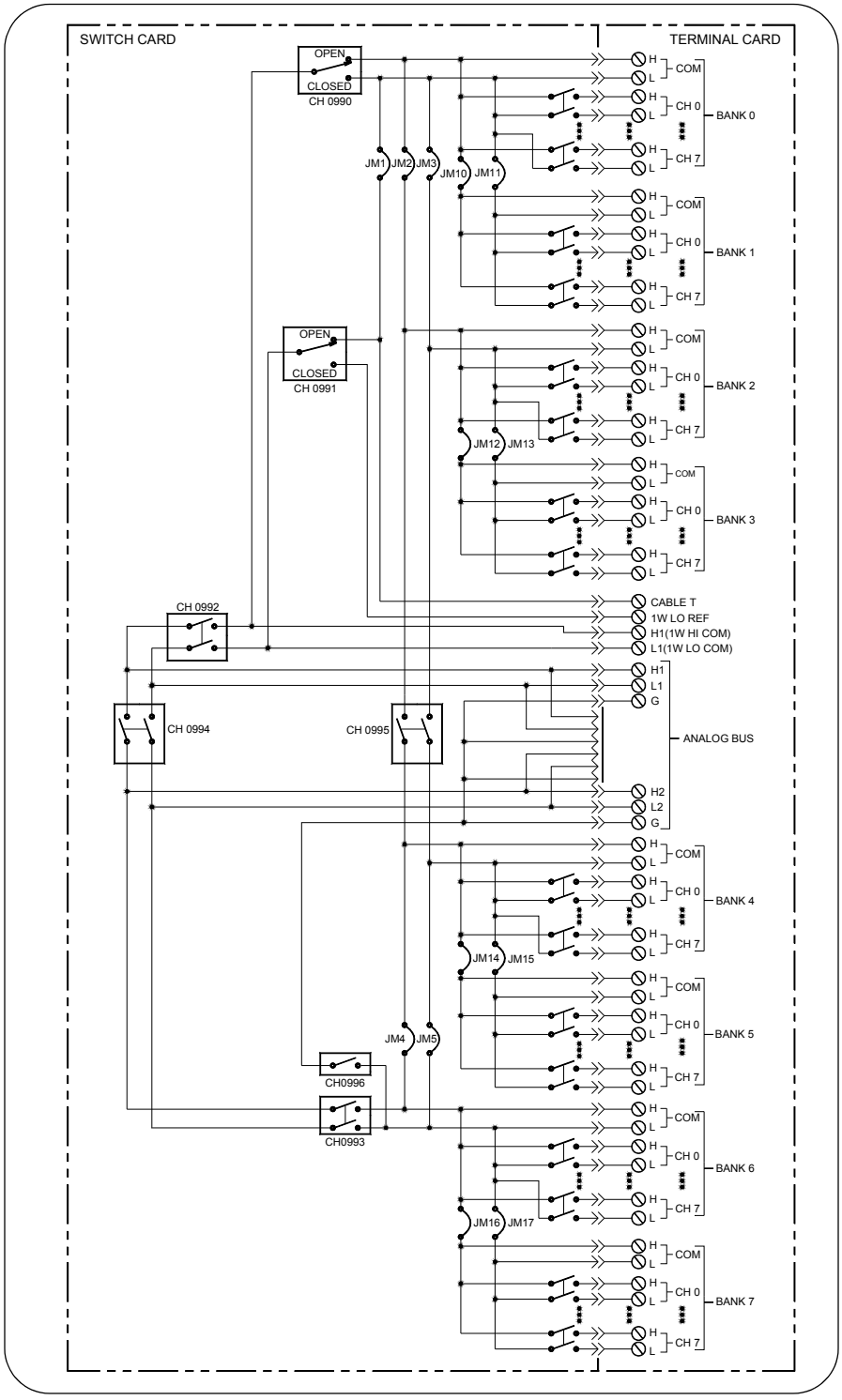


Figure 1-1. E1460A Multiplexer Block Diagram

Banks are arranged as follows:

- Bank 0 includes channels 00 through 07 and COM
- Bank 1 includes channels 10 through 17 and COM
- Bank 2 includes channels 20 through 27 and COM
- Bank 3 includes channels 30 through 37 and COM
- Bank 4 includes channels 40 through 47 and COM
- Bank 5 includes channels 50 through 57 and COM
- Bank 6 includes channels 60 through 67 and COM
- Bank 7 includes channels 70 through 77 and COM

Each channel is switched (connected to its common) by closing the appropriate (latching) relays. Channels 0 through 7 can be switched to COM for all banks. Any number of channels in each bank can be connected to common at a time (except for one-wire mode).

User inputs/outputs to each channel are via wire terminals. When a channel is closed, it is internally connected to the COM terminal. When a channel is opened, it is internally disconnected. Open channels are not terminated.

At power-on or reset, all channels are switched open (non-terminated) for all banks only when using the SCPI or C-SCPI driver. At power-off, all relays remain in their present state.

Control Relays

In addition to the channel switching relays, the switch card contains seven control relays (numbered 0990 to 0996). These relays switch the COM lines of banks dependent on the mode selected. All relays are automatically selected when the module is configured for the desired mode, when using the [ROUTe:]FUNCtion <*card_number*>, <*function*> command.

For the *stand-alone switchbox configuration*, this command must be used in conjunction with the following commands. If you only use [ROUTe:]OPEN and [ROUTe:]CLOSe commands, the appropriate control relays must also be closed with the CLOSe command.

```
[ROUTe:]SCAN:MODE mode
[ROUTe:]SCAN:PORT
[ROUTe:]SCAN channel_list
```

For the *scanning multimeter configuration,* [ROUTe:]FUNCtion <*card_number*>, <*function*> in conjunction with the CONFigure and INITiate or MEASure multimeter commands closes the appropriate control relays. See Chapter 3 in this manual and Chapter 5 in the *E1326B/E1411B User's Manual* for more information about these commands. Table 1-1 shows the control relay functions.

Table 1-1. Control Relay Functions

Control Relay	Function
0990	Selects HI or LO terminal for one-wire switching.
0991	Connects Cable Test or one-wire LO REF terminal to the one-wire LO COM terminal.
0992	Connects lower 32 channels (banks 0 - 3) to analog bus.
0993	Connects upper 32 channels (banks 4 - 7) to analog bus.
0994	Connects lower and upper analog buses together.
0995	Connects lower and upper common buses together (64-channel, two-wire operation).
0996	Connects analog bus Guard to the LO line, on the upper 32 channels (banks 4 to 7).

Basic Operating Modes

The E1460A uses the channel and control relays on the switch card to perform four basic operating modes: one-wire, two-wire, three-wire, or four-wire as shown. Connections to the analog bus (for multimeter connection) are provided on both the relay switch card and terminal module.

One-wire Mode

Switches either the HI or LO terminal of a channel in banks 0 through 7 to the one-wire HI COM terminal. One-wire LO COM is switched to the one-wire LO REF terminal. Only one channel can be switched (closed) at a time. A maximum of 128 one-wire channels can be switched. SCAN goes through all channel relay lows. Then, control relay 0990 switches and SCAN goes through all channel relay highs.

Two-wire Mode

Switches both the HI and LO terminals of a channel in banks 0 through 7 to the HI COM and LO COM terminals. A maximum of 64 two-wire channels can be switched.

Three-wire Mode

Switches both the HI and LO terminals of a channel in banks 0 through 3 to the HI COM and LO COM terminals. This mode also switches the LO terminal of the pair channel in banks 4 through 7 to the LO COM terminal. In addition, the low terminal of the pair channel in banks 4 through 7 can be connected to the analog bus Guard terminal. Banks are paired 0/4, 1/5, 2/6, and 3/7. A maximum of 32 three-wire channels can be switched.

Four-wire Mode

Switches both the HI and LO terminals of a channel in banks 0 through 3 to the HI COM and LO COM terminals. Also switches the HI and LO terminals of the pair channel in banks 4 through 7 to the HI COM and LO COM terminals. Banks are paired 0/4, 1/5, 2/6, and 3/7. A maximum of 32 four-wire channels can be switched.

Configuring the Multiplexer

This section gives guidelines to configure the relay switch card. See "Configuring Terminal Modules" for guidelines to configure the terminal modules. This section includes:

- Warnings and Cautions
- Setting the Logical Address Switch
- Setting the Status Register Switch
- Setting the Interrupt Priority
- Configuring the Switch Card Wire Jumpers
- Installing the Multiplexer in a Mainframe
- Connecting the Analog Bus

Warnings and Cautions

WARNING **SHOCK HAZARD. Only service-trained personnel who are aware of the hazards involved should install, remove, or configure the multiplexer. Before you remove any installed module, disconnect AC power from the mainframe and from other modules that may be connected to the multiplexer.**

WARNING **CHANNEL WIRING INSULATION. All channels that have a common connection must be insulated so that the user is protected from electrical shock in the event that two or more channels are connected together. This means wiring for all channels must be insulated as though each channel carries the voltage of the highest voltage channel.**

CAUTION **MAXIMUM INPUTS.** The maximum voltage that can be applied to any terminal is 220 Vdc/250 Vrms. The maximum current that can be applied to any terminal is 1A at 30 Vdc/Vrms, or 0.3 A at 250 Vdc/Vrms. The maximum power that can be applied to any terminal is 40 VA.

CAUTION **STATIC ELECTRICITY.** Static electricity is a major cause of component failure. To prevent damage to the electrical components in the multiplexer, observe anti-static techniques whenever removing a module from the mainframe or whenever working on a module.

Setting the Logical Address Switch

Plug-in modules installed in an mainframe or used with a command module are treated as independent *instruments* each having a unique secondary GPIB address. Each instrument is also assigned a dedicated error queue, input and output buffers, status registers and, if applicable, dedicated mainframe/command module memory space for readings or data. An instrument may be composed of a single plug-in module (such as a counter) or multiple plug-in modules (for a switchbox or scanning multimeter instrument).

The instrument logical address (LADDR) is set with the logical address switch located on the instrument. The logical address switch (LADDR) factory setting for the E1460A is 112. Valid address values are from 1 to 255. See Figure 1-2 to set the logical address. From Figure 1-2, note that the value of the logical address set is the sum of the values of the switches set to the CLOSED position.

NOTE *The address switch selected value must be a multiple of 8 if the module is the first module in a switchbox used with a VXIbus command module and being instructed by SCPI commands.*

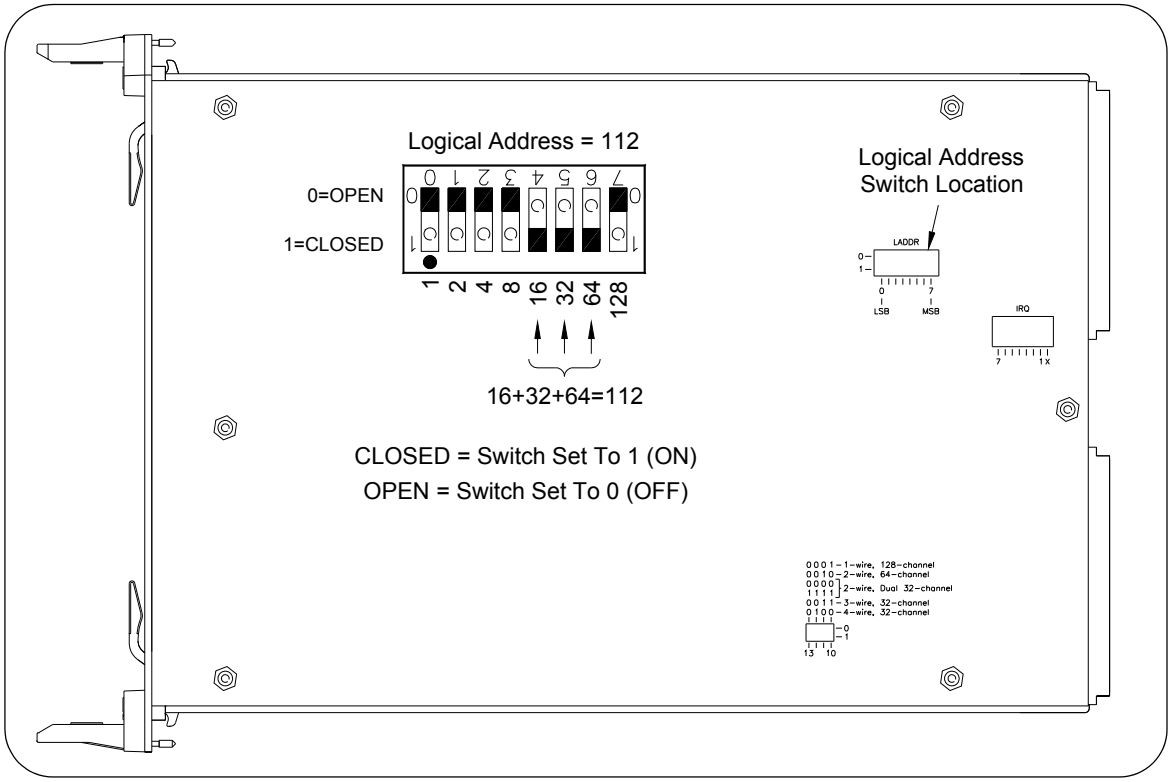


Figure 1-2. Setting the Logical Address Switch

Setting the Status Register Switch

Four bits of the Status Register (bits 10-13) define whether the multiplexer module is set for one-wire, two-wire, three-wire, or four-wire switching. To ensure proper operation, set the status register switch as shown in Figure 1-3.

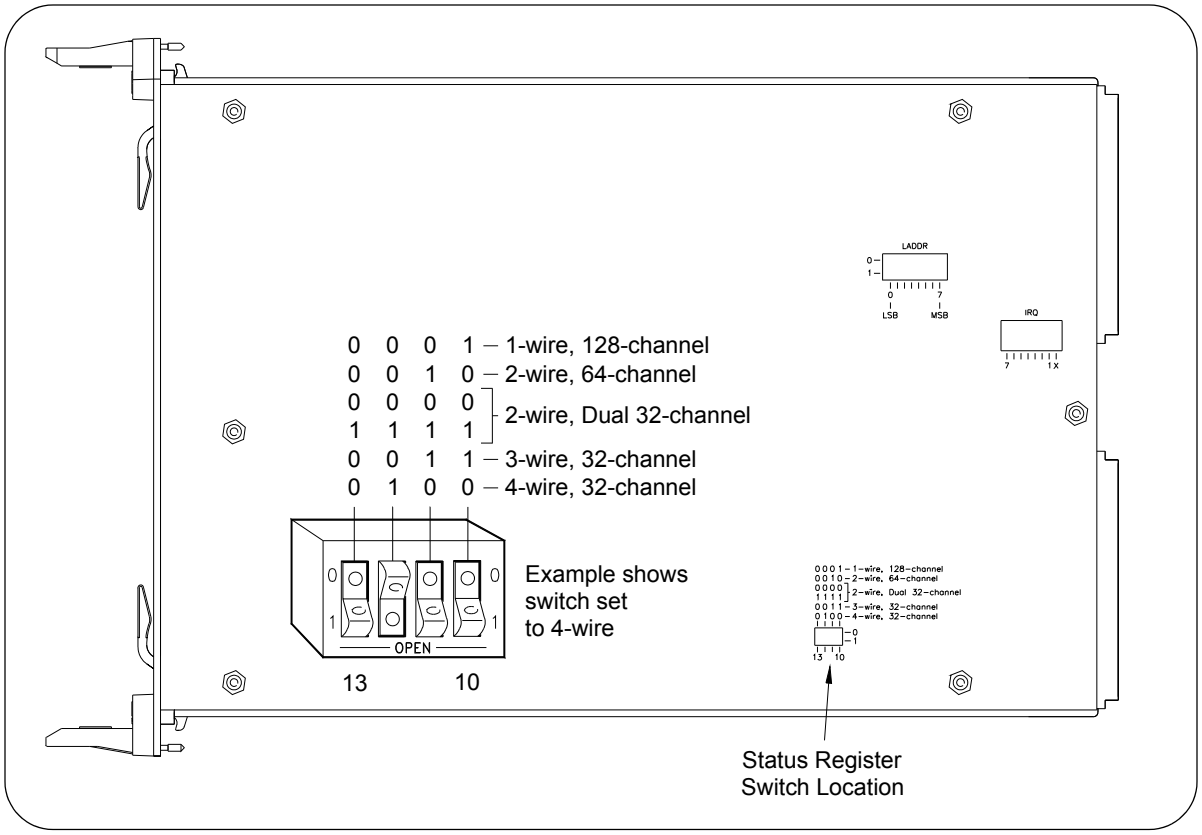


Figure 1-3. Setting the Status Register Switch

Setting the Interrupt Priority

The multiplexer module generates an interrupt after a channel has been closed. These interrupts are sent to, and acknowledgments are received from, the command module (such as an E1406A) via the VXIbus backplane interrupt lines.

For most applications where the multiplexer module is installed in a C-Size mainframe, the interrupt priority jumper does not have to be moved. This is because the VXIbus interrupt lines have the same priority, and interrupt priority is established by installing modules in slots numerically closest to the command module. Thus, slot 1 has a higher priority than slot 2, slot 2 has a higher priority than slot 3, etc.

See Figure 1-4 to change the interrupt priority. You can select eight different interrupt priority levels. Level 1 is the lowest priority and Level 7 is the highest priority. Level X disables the interrupt. The module's factory setting is Level 1. To change, remove the 4-pin jumper from the old priority location and reinstall in the new priority location. If the 4-pin jumper is not used, the two jumper locations must have the same interrupt priority level selected.

NOTE *The interrupt priority jumper MUST be installed in position 1 when using the E1406 command module. Level X interrupt priority should not be used under normal operating conditions. Changing the priority level jumper is not recommended. Do not change unless specifically instructed to do so.*

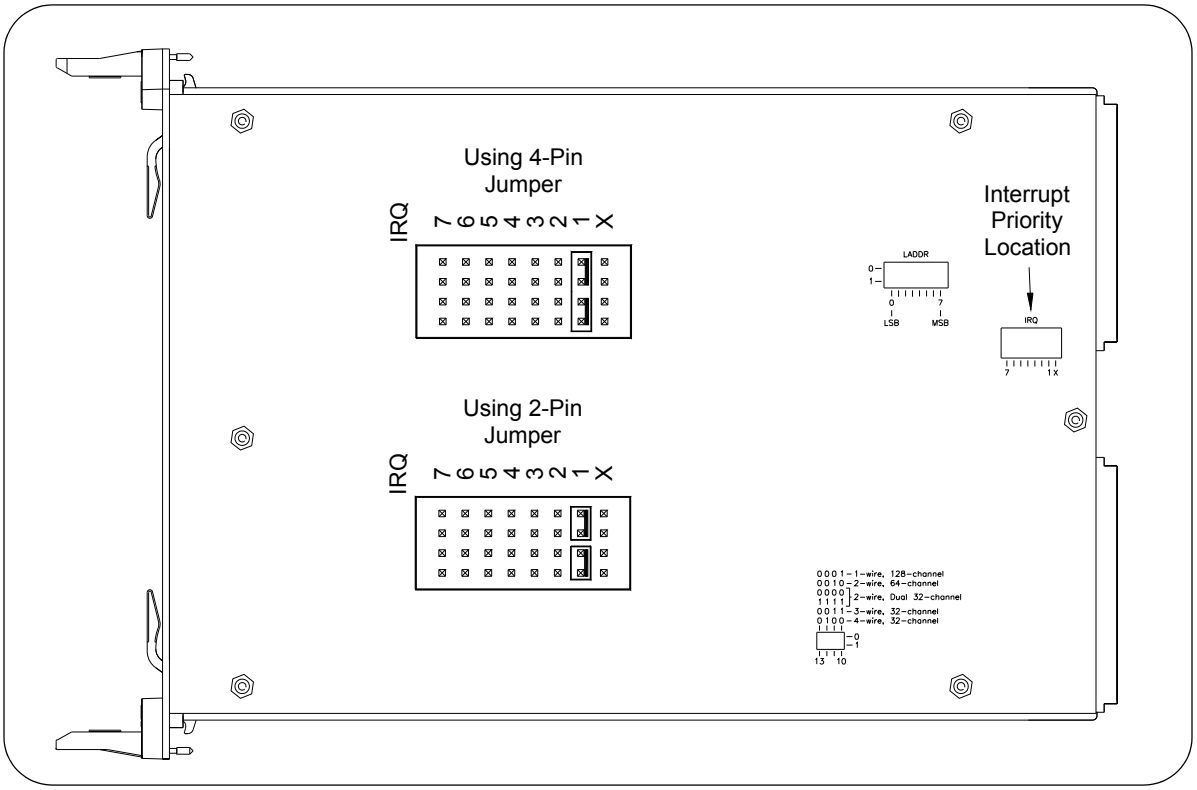


Figure 1-4. Setting the Interrupt Priority

Configuring the Switch Card Wire Jumpers

The relay switch card has thirteen factory-installed wire jumpers (see Figures 1-1 and 1-5) that connect COM lines of banks together to form dual 1x32 channel configurations. These wire jumpers can be changed to reconfigure the switch card to various 8-channel or 16-channel configurations.

NOTE *It is only necessary to change the wire jumpers when reconfiguring the switch card for groups of eight or 16 channels (from 32). DO NOT CHANGE the wire jumper positions unless instructed to do so in the applicable operating procedures.*

Wire Jumper Functions

With the exception of JM1, wire jumpers are changed in pairs. Functions of the wire jumpers are:

- JM1: Used during cable test (see Chapter 2)
- JM2/JM3: Used to connect the COM lines of bank pairs 0/1 and 2/3
- JM4/JM5: Used to connect the COM lines of bank pairs 4/5 and 6/7
- JM10/JM11: Used to connect the COM lines of banks 0 and 1
- JM12/JM13: Used to connect the COM lines of banks 2 and 3
- JM14/JM15: Used to connect the COM lines of banks 4 and 5
- JM16/JM17: Used to connect the COM lines of banks 6 and 7

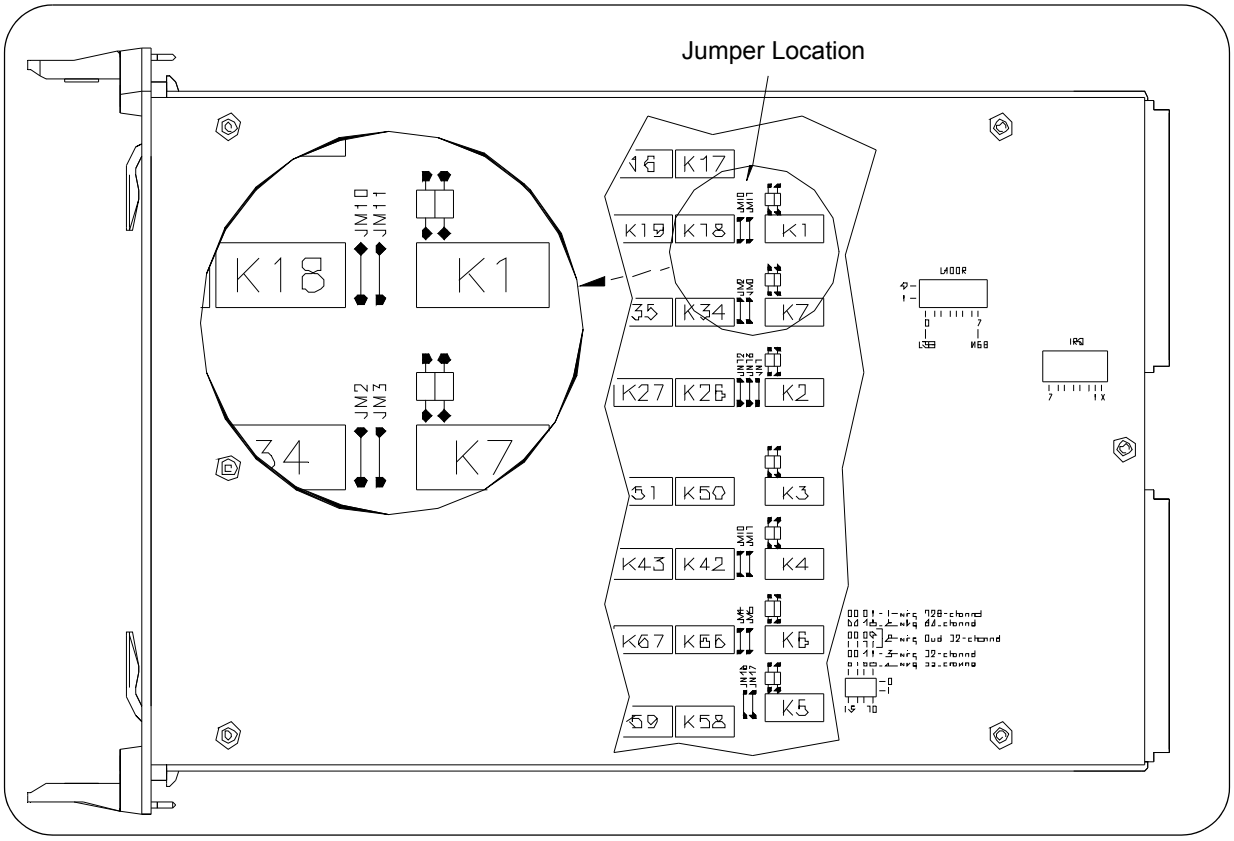


Figure 1-5. Switch Card Wire Jumper Settings

Reconfiguring the Relay Switch

To reconfigure the relay switch card:

1 Position the switch card on a flat surface. Using a TORX T-10 driver, remove the six screws on the shield and carefully lift the shield to expose the printed circuit board.

2 Configure the wire jumpers as required using Table 1-2. If you install new jumpers, use zero-ohm resistors or No. 22 AWG copper wire.

For example, to configure banks 0, 1, 2, and 3 as 1x8 multiplexers and banks 4, 5, 6, and 7 as 1x16 multiplexers, jumper positions are: Jumpers = JM14,15,16,17 and No Jumpers = JM2,3,4,5,10,11,12,13.

3 Replace the shield and re-install the six screws.

NOTE *When wire jumpers JM10 through JM17 are removed, the odd-numbered banks can no longer be connected to the analog bus. For example, if JM10 and JM11 are removed, then bank 1 can no longer be connected to the analog bus terminals (except through user wiring).*

When wire jumpers JM2 through JM5 are removed, banks 2/3 and 4/5, respectively, can no longer be connected to the analog bus. For example, if JM2 and JM3 are removed, then banks 2 and 3 can no longer be connected to the analog bus terminals (except through user wiring).

Table 1-2. Jumper Configurations

Bank Number = Jumper Configuration				JM Number (0 = Jumper, 1 = No Jumper)												
Bank 0	Bank 1	Bank 2	Bank 3	1	2	3	4	5	10	11	12	13	14	15	16	17
1x32*	1x32*	1x32*	1x32*	-	0	0	-	-	0	0	0	0	-	-	-	-
1x16	1x16	1x16	1x16	-	1	1	-	-	0	0	0	0	-	-	-	-
1x8	1x8	1x8	1x8	-	1	1	-	-	1	1	1	1	-	-	-	-
1x8	1x8	1x16	1x16	-	1	1	-	-	1	1	0	0	-	-	-	-
1x16	1x16	1x8	1x8	-	1	1	-	-	0	0	1	1	-	-	-	-

Bank Number = Jumper Configuration				JM Number (0 = Jumper, 1 = No Jumper)												
Bank 4	Bank 5	Bank 6	Bank 7	1	2	3	4	5	10	11	12	13	14	15	16	17
1x32*	1x32*	1x32*	1x32*	-	-	-	0	0	-	-	-	-	0	0	0	0
1x16	1x16	1x16	1x16	-	-	-	1	1	-	-	-	-	0	0	0	0
1x8	1x8	1x8	1x8	-	-	-	1	1	-	-	-	-	1	1	1	1
1x8	1x8	1x16	1x16	-	-	-	1	1	-	-	-	-	1	1	0	0
1x16	1x16	1x8	1x8	-	-	-	1	1	-	-	-	-	0	0	1	1

* factory setting

Installing the Multiplexer in a Mainframe

The E1460A can be installed in any slot (except slot 0) in a C-Size VXIbus mainframe. See Figure 1-6 to install the multiplexer in a mainframe.

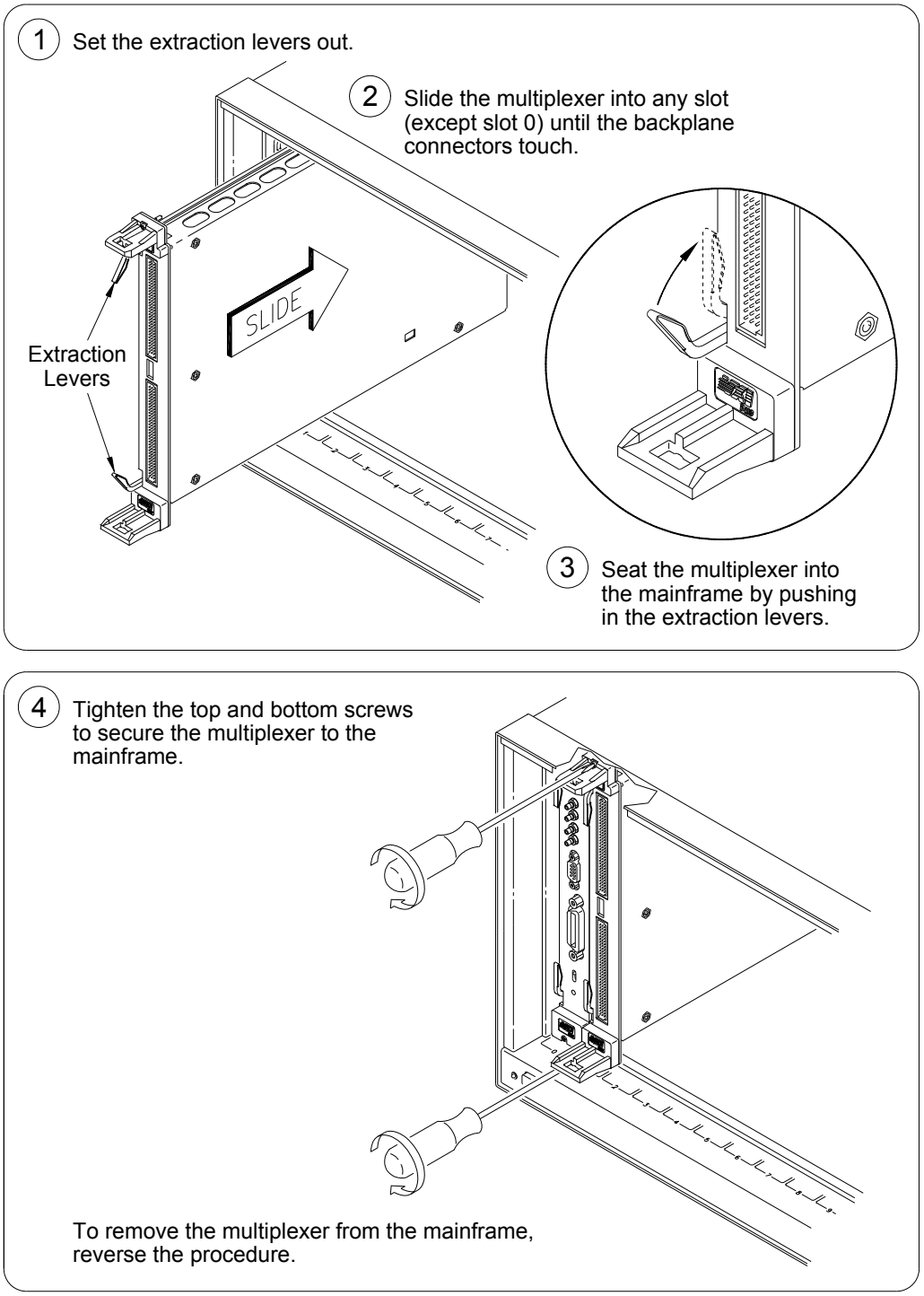

1 Set the extraction levers out.
2 Slide the multiplexer into any slot (except slot 0) until the backplane connectors touch.
SLIDE
Extraction Levers
3 Seat the multiplexer into the mainframe by pushing in the extraction levers.
4 Tighten the top and bottom screws to secure the multiplexer to the mainframe.
To remove the multiplexer from the mainframe, reverse the procedure.

Figure 1-6. Installing the Multiplexer in a VXIbus Mainframe

Connecting the Analog Bus

Figure 1-7 shows how to connect the analog bus between multiple multiplexer modules and to the E1411B multimeter. Use cable (part number E1400-61605) to connect the analog bus to all the modules.

NOTE *The analog bus can also be wired to the terminal module. See "Standard Terminal Module Description" for more information.*

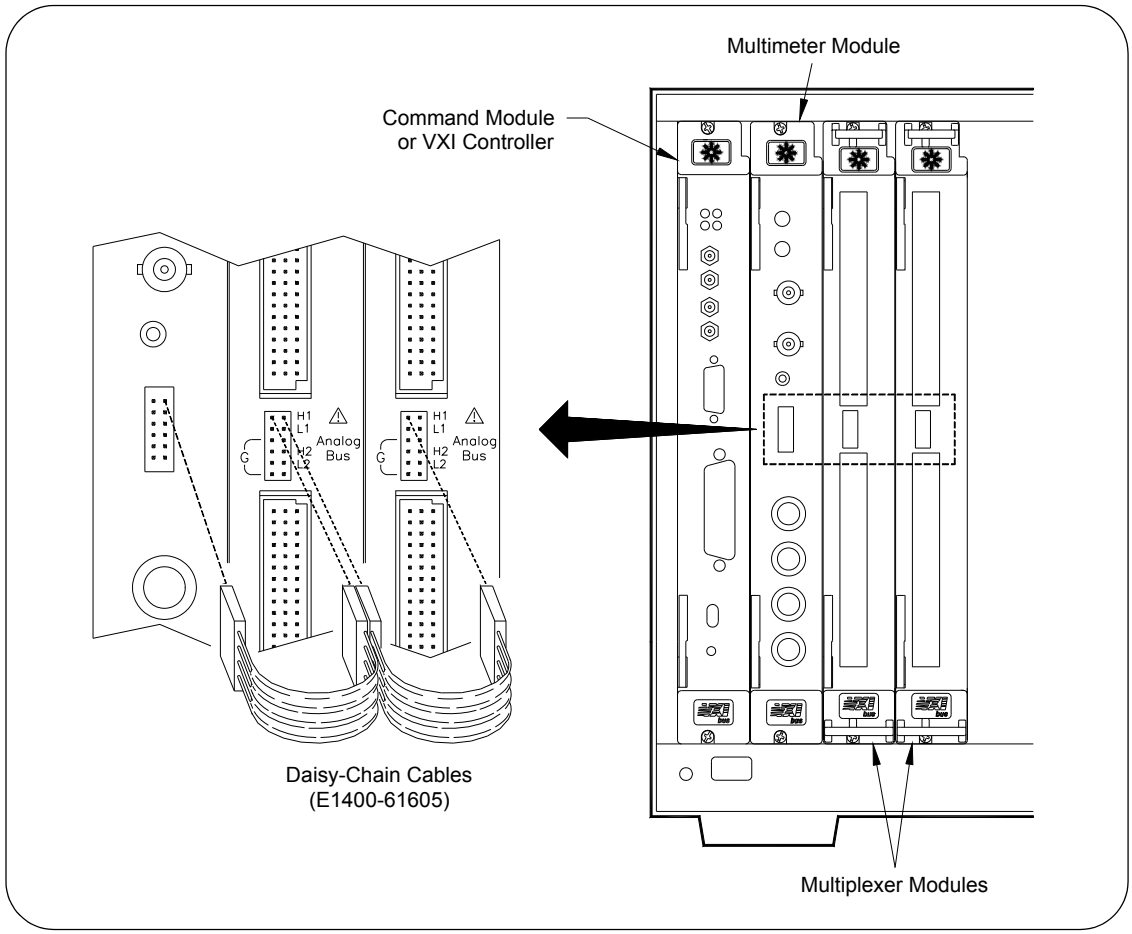


Figure 1-7. Analog Bus Cable Connections

Configuring Terminal Modules

The E1460A 64-Channel Relay Multiplexer consists of a relay switch card and a (standard) screw-type terminal module or a crimp-and-insert terminal module (Option A3E). See Figure 1-10 for the multiplexer's connector pin-out that mates to the terminal module.

Standard Terminal Module Description

Figure 1-8 shows the standard screw-type terminal module connectors and associated bank numbers, channel numbers, and line designations. Use the following guidelines for wiring connections:

- Be sure that wires make good connections on screw terminals.
- Maximum terminal wire size is No. 16 AWG. When wiring all 64-channels, a smaller gauge wire (20-22 AWG) is recommended.
- Wire ends should be stripped 6mm (0.25 in.) and tinned to prevent single strands from shorting to adjacent terminals.

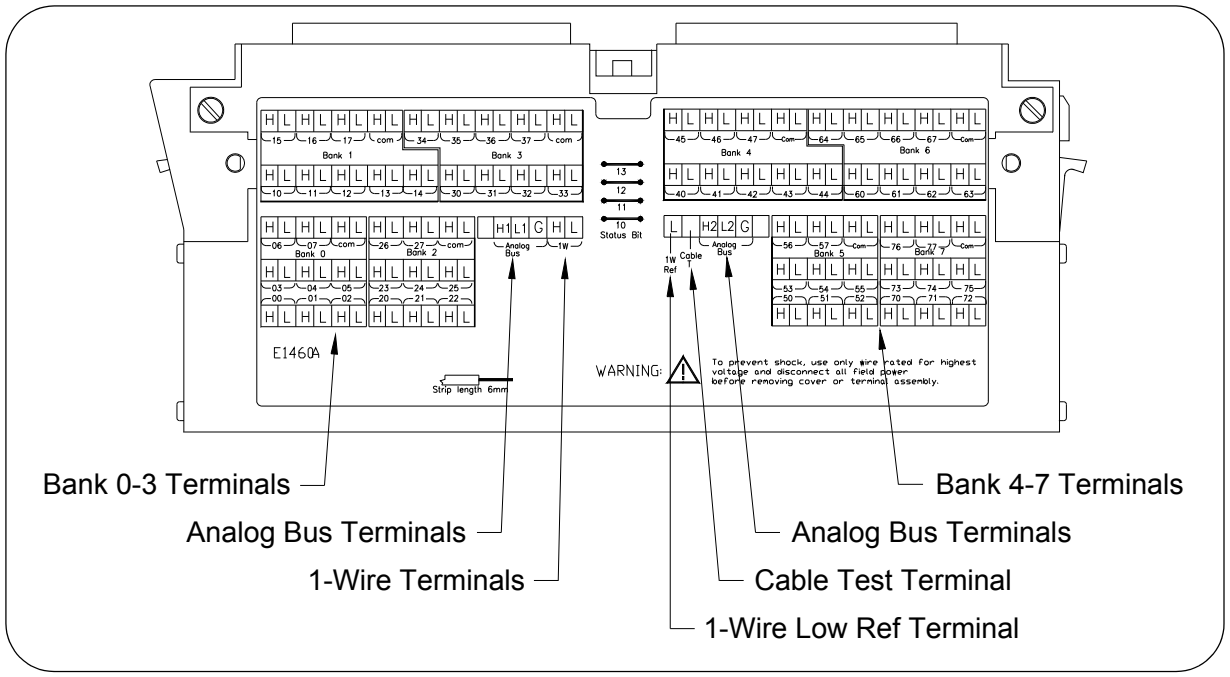


Figure 1-8. Standard Screw-type Terminal Module

Terminal Module Option A3E Description

Terminal module Option A3E (see Figure 1-9) provides a crimp-and-insert terminal module that allows you to crimp connectors onto wires which are then inserted directly into the multiplexer's mating connector. See the pin-out diagram (Figure 1-10) to make the connections. Table 1-3 shows the accessories that can be used with crimp-and-insert Option A3E.

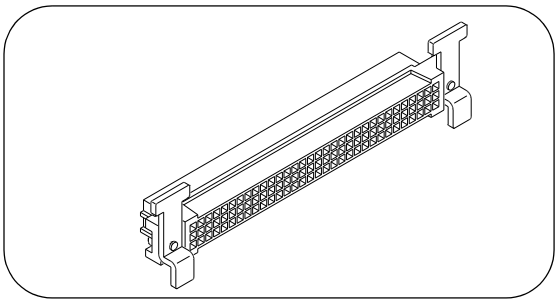
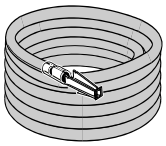
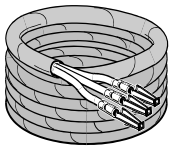
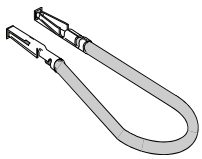
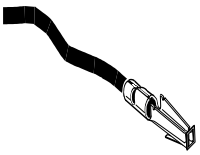


Figure 1-9. Option A3E Crimp-and-Insert Connector

Table 1-3. Option A3E Terminal Module Accessories

Accessory	Description	Picture	Specifications
Single-Conductor and Contact	A crimp-and-insert contact is crimped onto one end of a wire. The other end is not terminated. Order 91510A.		Length: 2 meters Wire Gauge: 24 AWG Quantity: 50 each Insulation Rating: 105°C max Voltage: 300 V
Shielded-Twisted-Pair and Contacts	A crimp-and-insert contact is crimped onto each conductor at one end of a shielded-twisted-pair cable. The other end is not terminated. Order 91511A.		Length: 2 meters Wire Gauge: 24 AWG Outside Diameter: 0.1 inches Quantity: 25 each Insulation Rating: 250°C max Voltage: 600 V
Jumper Wire and Contacts	A crimp-and-insert contact is crimped onto each end of a single conductor jumper wire. This jumper is typically used to tie two pins together in a single crimp-and-insert connector. Order 91512A.		Length: 10 cm Wire Gauge: 24 AWG Quantity: 10 each Insulation Rating: 105°C max Voltage: 300 V
Crimp-and-Insert Contacts	These contacts may be crimped onto a conductor and then inserted into a crimp-and-insert connector. The crimp tool kit is required to crimp the contacts onto a conductor and remove the contact from the connector. Order 91515A.		Wire Gauge Range: 20-26 AWG Quantity: 250 each Plating: Gold Plated Contact Maximum Current: 2A at 70°C
Crimp-and-Insert Tools	The hand crimp tool (part number 91518A) is used for crimping contacts onto a conductor. The pin extractor tool (part number 91519A) is required for removing contacts from the crimp-and-insert connector. *These products are not included with Option A3E or with the terminal option accessories listed earlier.*		
Extra Crimp-and-Insert Connectors	The crimp-and-insert connector is normally supplied with Option A3E. Contact Agilent if additional connectors are needed. Order 91484B.		

Connecting User Inputs

Figure 1-10 shows the front panel of the E1460A and the multiplexer's connector pin-out which mates to the terminal module. Actual user inputs are connected to the terminal module. See "Wiring Terminal Modules" for connection information.

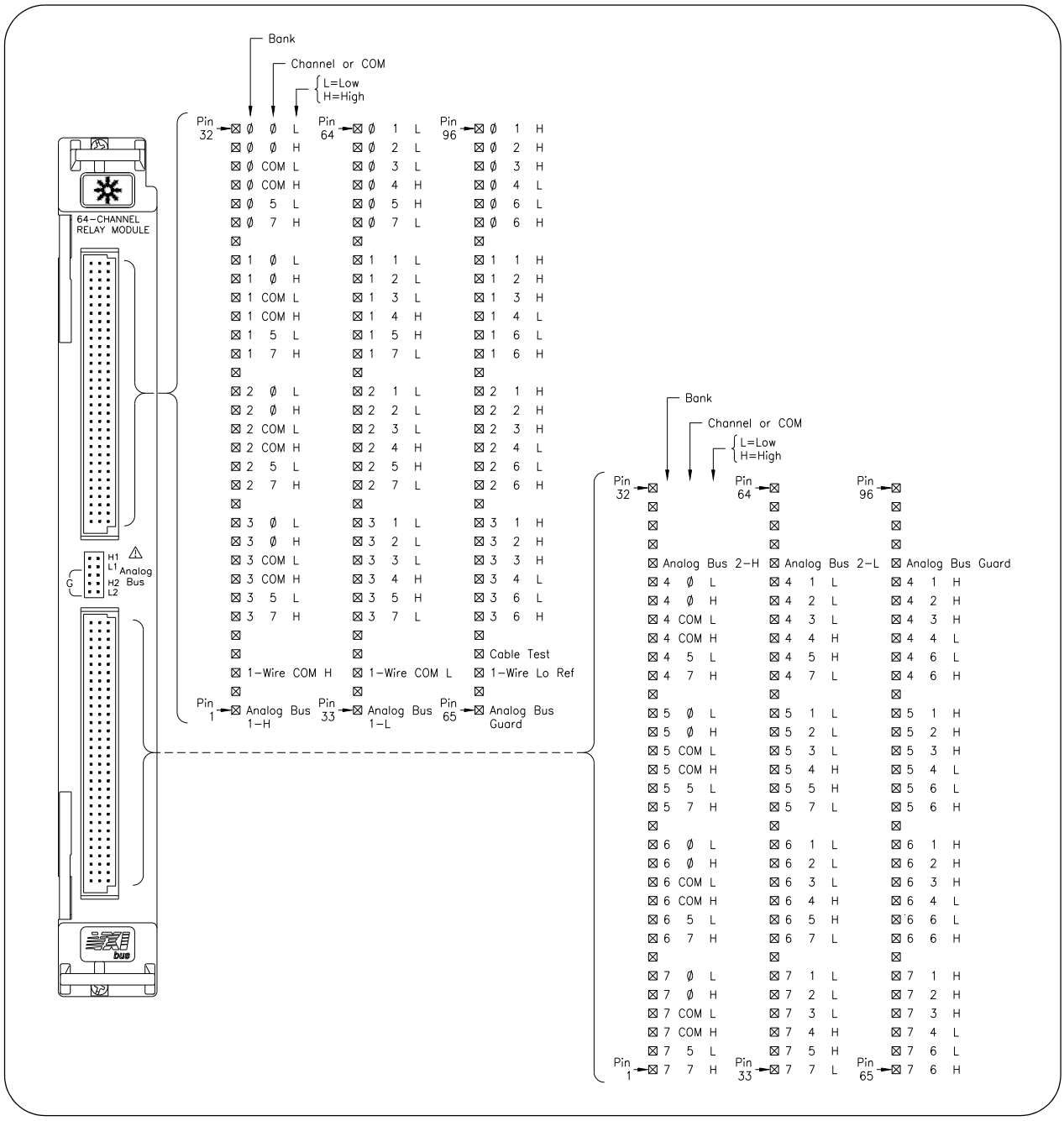


Figure 1-10. E1460A Multiplexer Pin-Out

Wiring Terminal Modules

Figures 1-11 and 1-12 show suggested steps to connect field wiring (user inputs) to a terminal module.

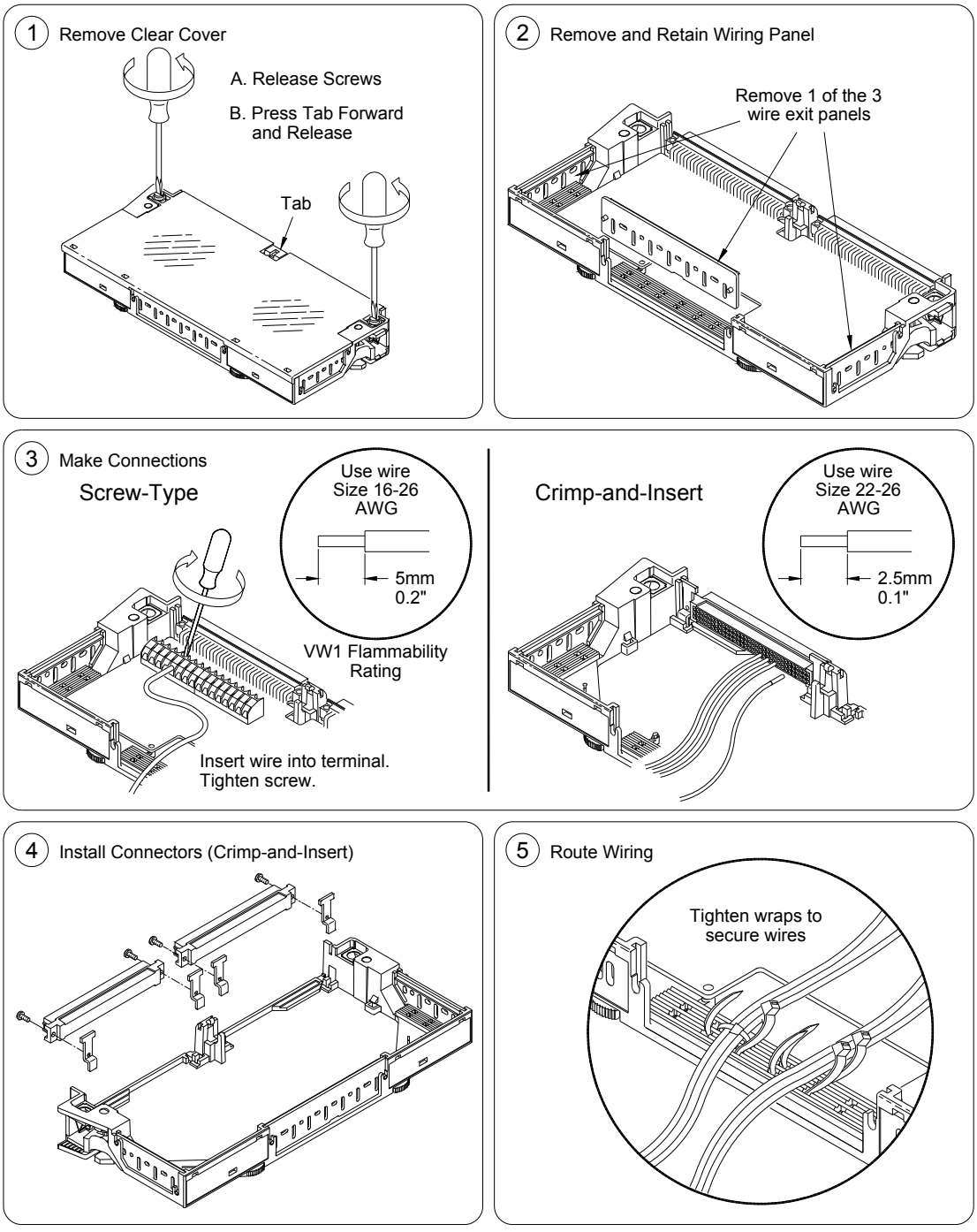


Figure 1-11. Steps to Wire Terminal Modules

Continued on next page

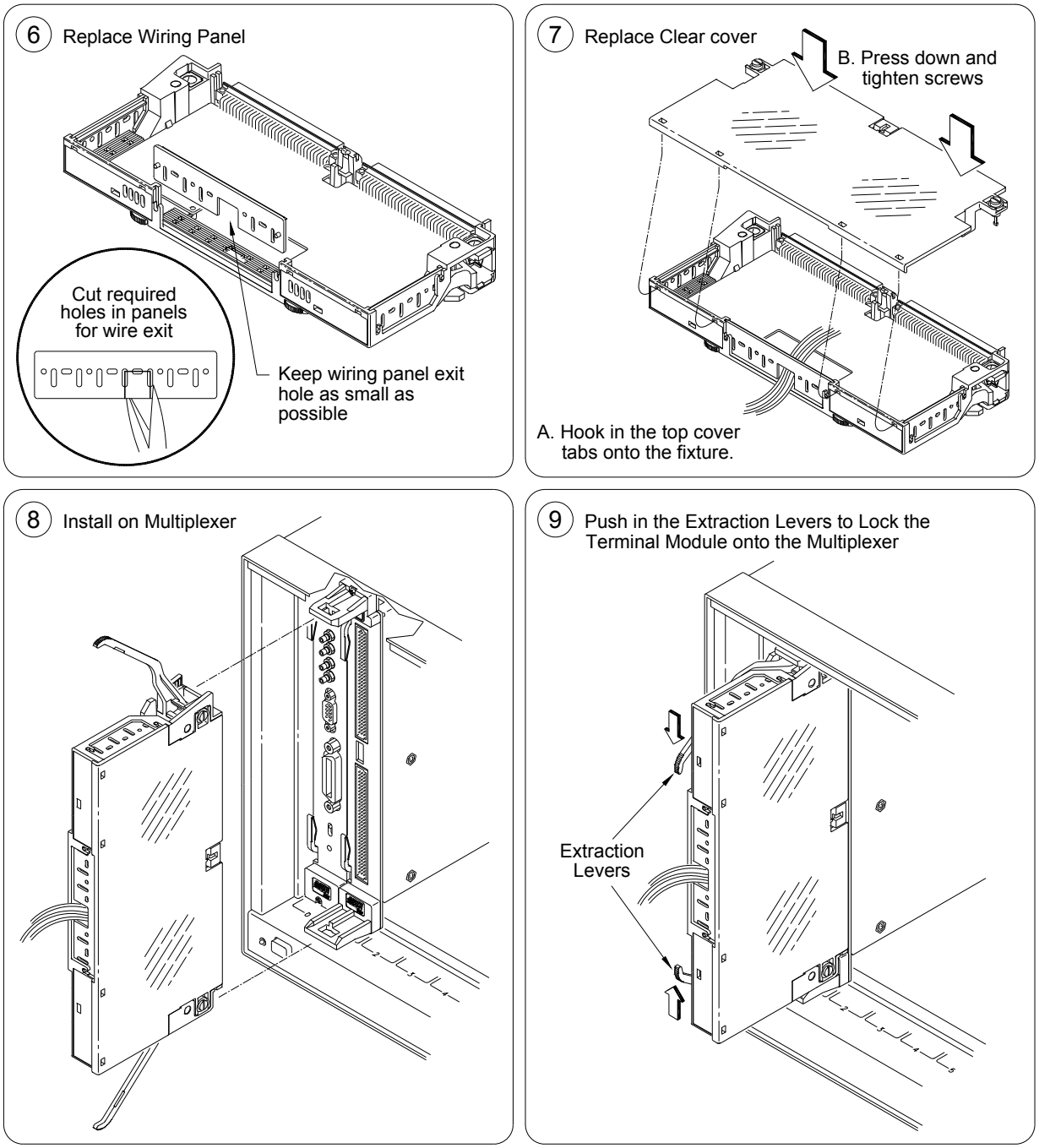

6 Replace Wiring Panel
Cut required holes in panels for wire exit
Keep wiring panel exit hole as small as possible
7 Replace Clear cover
B. Press down and tighten screws
A. Hook in the top cover tabs onto the fixture.
8 Install on Multiplexer
9 Push in the Extraction Levers to Lock the Terminal Module onto the Multiplexer
Extraction Levers

Figure 1-12. Steps to Wire Terminal Modules (continued)

Attaching Terminal Modules to the Multiplexer

Figure 1-13 shows how to attach a terminal module to the multiplexer and how to remove a terminal module from the multiplexer.

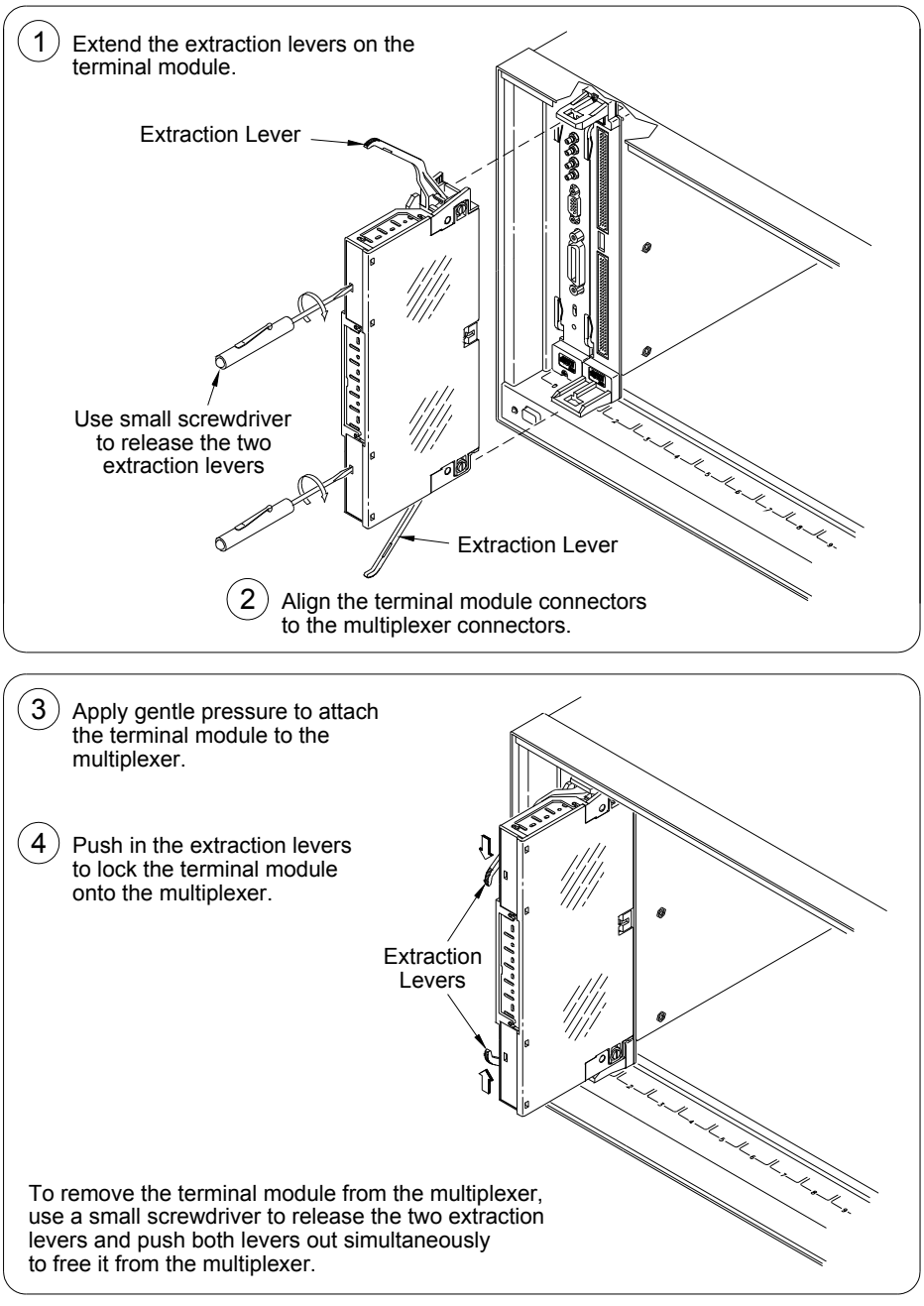

1
Extend the extraction levers on the terminal module.
Extraction Lever
Use small screwdriver to release the two extraction levers
Extraction Lever
2
Align the terminal module connectors to the multiplexer connectors.
3
Apply gentle pressure to attach the terminal module to the multiplexer.
4
Push in the extraction levers to lock the terminal module onto the multiplexer.
Extraction Levers
To remove the terminal module from the multiplexer, use a small screwdriver to release the two extraction levers and push both levers out simultaneously to free it from the multiplexer.

Figure 1-13. Attaching a Terminal Module to the Multiplexer

Programming the Multiplexer

The multiplexer modules are programmed using either a switchbox or scanning multimeter configuration. To program the multiplexer modules using SCPI commands, you must choose the controller language, interface address, and SCPI commands to be used. Guidelines to choose SCPI commands for the multiplexer follow.

NOTE *This discussion applies only to SCPI programming. See Appendix B - Register-Based Programming for details on multiplexer module registers.*

Checking SCPI Drivers

The E1460A operates with Switchbox Driver Revision A.08.03 or later or with Scanning Voltmeter Driver Revision A.06.03 or later. The E1460A may be recognized by earlier driver revisions, but will not operate properly. Before using the E1406A, you should check your driver revision and, if necessary, load a new driver.

This procedure shows a way to download SCPI drivers to the E1406A. SCPI Instrument Drivers and the VXI Installation Consultant (VIC) are on the *Agilent Technologies Universal Instrument Drivers* CD. For the latest information on drivers, see the Agilent web site:

`http://www.agilent.com/find/inst_drivers`

What are SCPI Device Drivers?

Agilent register-based modules are supported by Standard Commands for Programmable Instruments (SCPI) drivers. These drivers reside in E1406A Command Module non-volatile memory. If you add a new register-based module to an existing VXI system and plan to program the module using SCPI, the firmware in your command module may need to be upgraded to accommodate the new module. You can download new drivers into non-volatile memory from controllers running Windows, BASIC, or IBASIC.

Checking the SCPI Driver Revision

This procedure describes how to decide which E1460A driver to use, how to check the currently installed driver, and how to determine if you need to download a new driver. If you determine that you need to install a new driver, see "Downloading a New Driver".

1. Decide whether to use the VOLTMTR or SWITCH driver. Use the VOLTMTR driver if you intend to use the E1460A in combination with the E1326B or E1411B multimeter in a Scanning Voltmeter configuration. In this configuration, the E1460A scans measurement channels and sends the signals to the multimeter where the measurements take place. Use the SWITCH driver for all other applications (all non-Scanning Voltmeter applications).

2. Check the currently installed driver revision numbers by sending the DIAG:DRIV:LIST? command to the command module (the command module is usually at GPIB address 70900). A typical result follows. The specific result depends on the specific drivers previously loaded into your command module.

SYSTEM,E1406A,A.08.00,ROM;IBASIC,IBASIC,A.O4.02,ROM;
VOLTMTR,E1326B,A.06.00,ROM;
SWITCH,SWITCHBOX,A.07.00,ROM;
COUNTER,E1332A,A.04.02,ROM;E1333A,A.04.02,ROM;
DIG_I/O,E1330A,A.O4.03,ROM;D/A,E1328A,A.04.02,ROM

3 Determine whether to install a new driver. The E1460A requires a SWITCH Driver Revision of A.08.03 or later or a VOLTMTR Driver Revision A.06.03 or later. In the example response above, the currently installed drivers are:

VOLTMTR,E1326A,A.06.00,ROM
SWITCH,SWITCHBOX,A.07.00,ROM

In this example, you must download a new SWITCH or VOLTMTR driver (depending upon which driver you chose in Step 1).

Downloading a New Driver

To download a new driver, choose your operating system and interface from the following list and follow the related instructions.

Windows via GPIB or RS-232. (For the fastest download, use GPIB rather than RS-232.) Use the *VXI Installation Consultant* (*VIC*). VIC is a hardware installation program that helps you configure and install VXI instruments and can also download DOS-formatted instrument drivers. VIC downloads drivers during the configuration process and stores a copy of the driver in the `C:\VIC\DRIVERS` directory the FIRST TIME the instrument is configured.

NOTE *If you are updating an already installed driver, the new driver must be downloaded using the VIC Driver Download utility. Instructions for using VIC and its Driver Download utility are contained in VIC's on-line help.*

All other operating systems/interfaces. See the *Installing SCPI Device Drivers Installation Note* (shipped with the downloadable drivers).

Multiplexer Addressing

To address specific channels within a multiplexer module in either switchbox or scanning multimeter configuration, you must end the appropriate SCPI command string to the switchbox or scanning multimeter (for example, CLOSe, OPEN, etc.) and specify the specific channel address.

SCPI Commands Format

You can send SCPI commands in either short or long form. A long form example is CLOSe(@123). The same command shown without the lower case letters is the short form. The command then becomes CLOS(@123).

Some commands are shown with brackets ([ ]). These are implied commands that you do not need to execute. The brackets are not part of the command and are not sent to the instrument.

For example, the ROUTe command is an implied command and is shown here as [ROUTe:]CLOS(@123). Thus, to execute these commands, you can just enter CLOS(@123). See Chapter 3 for more information about SCPI commands and how to send them.

Multiplexer Card Numbers

The multiplexer *card number* identifies the module within a switchbox or scanning multimeter configuration. The card number assigned depends on the configuration. Leading zeroes can be ignored for the card number.

Switchbox Configuration. In a single-module switchbox configuration, the card number is always 01. In a multiple-module switchbox configuration, multiplexer modules are set to successive logical addresses. The multiplexer module with the lowest logical address is always card number 01. The card number with the next successive logical address is 02, etc.. See Figure 1-14 for card numbers and logical addresses of a typical multiple-module switchbox configuration.

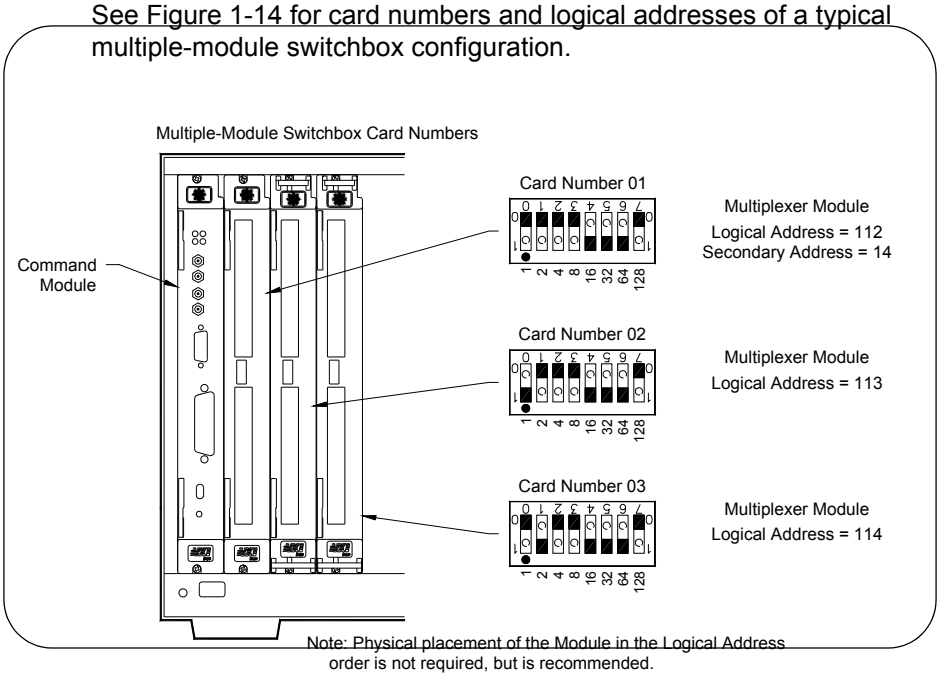

Multiple-Module Switchbox Card Numbers
Command Module
Card Number 01
Multiplexer Module
Logical Address = 112
Secondary Address = 14
Card Number 02
Multiplexer Module
Logical Address = 113
Card Number 03
Multiplexer Module
Logical Address = 114
Note: Physical placement of the Module in the Logical Address order is not required, but is recommended.

Figure 1-14. Card Numbers in a Multiple-Module Switchbox

Scanning Multimeter Configuration. In a multiple-module scanning multimeter configuration, modules are assigned successive logical addresses beginning with the multimeter. The multimeter module is always card number 00, the multiplexer module with the next lowest logical address is always card number 01, the next successive logical address is card number 02, etc. See Figure 1-15 for card numbers and logical addresses of a typical multiple-module scanning multimeter configuration.

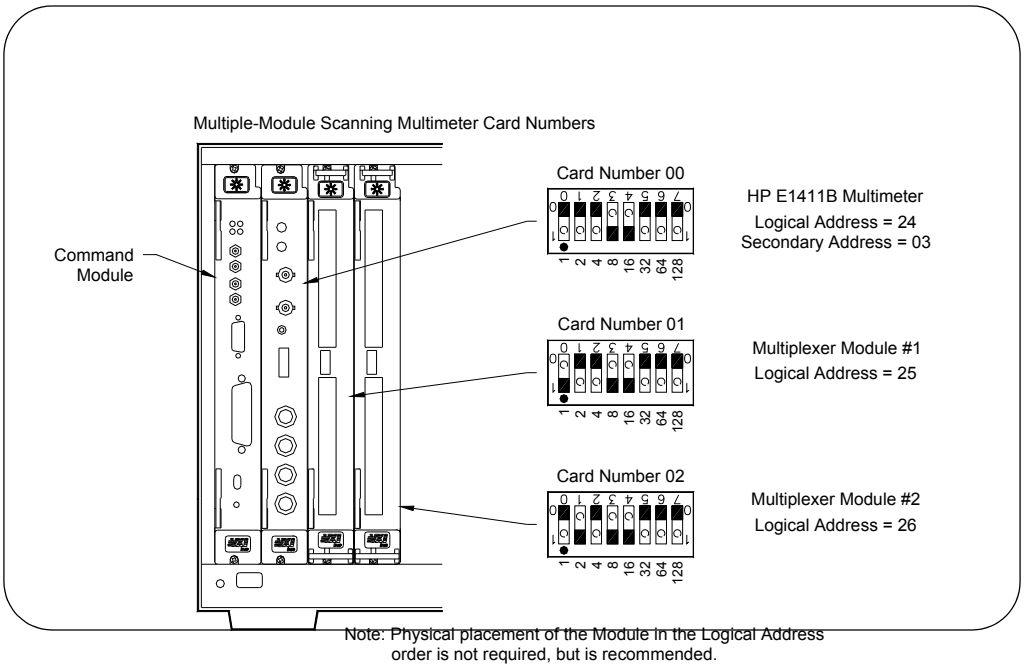

Multiple-Module Scanning Multimeter Card Numbers
Command Module
Card Number 00
HP E1411B Multimeter
Logical Address = 24
Secondary Address = 03
Card Number 01
Multiplexer Module #1
Logical Address = 25
Card Number 02
Multiplexer Module #2
Logical Address = 26
Note: Physical placement of the Module in the Logical Address order is not required, but is recommended.

Figure 1-15. Card Numbers in a Multiple-module Scanning Multimeter

Multiplexer Channel Addresses

For the E1460A, the channel address (*channel_list*) has the form:

(@*ssbc*) for two-wire, three-wire, and four-wire operating modes
(@*ss0hbc*) for one-wire operating mode
(@*ss099c*) for control relays (all operating modes)

where
ss = card number (01-99)
0h = LO or HI terminal (0-1)
b = bank number (0-7)
c = number 0-7 for switching relays or 0-6 for control relays

Channel addresses can be specified in the following forms. The leading zero in the card number can be ignored.

One-wire mode only

(@*ss0hbc*) for a single channel;
(@*ss0hbc,ss0hbc*) for multiple channels;
(@*ss0hbc:ss0hbc*) for sequential channels;
(@*ss0hbc:ss0hbc,ss0hbc:ss0hbc*) for groups of sequential channels or any combination of the above.

Two-wire, three-wire, or four-wire modes (and control relays) where *b* = 099

(@*ssbc*) for a single channel;
(@*ssbc,ssbc*) for multiple channels;
(@*ssbc:ssbc*) for sequential channels;
(@*ssbc:ssbc,ssbc:ssbc*) for groups of sequential channels;

or any combination of the above.

Low or High Terminal Number

The LO or HI terminal number is specified for one-wire mode only and identifies what terminal will be used during one-wire switching. This number can be omitted when the low terminal is the desired selection. Only valid terminals can be accessed in a channel list. 00 is specified to use the LO (L) terminal of the bank and channel selected. Defaults to LO terminal if not entered. 01 is specified to use the HI (H) terminal of the bank and channel selected.

Bank Number

The bank number identifies what bank of eight channels will be affected during switching. The bank numbers are 0 to 7 for one-wire and two-wire modes and 0 to 3 for three-wire and four-wire modes. Only valid banks can be accessed in a channel list. Closing, opening, or querying banks 4 to 7 when operating in three-wire and four-wire modes will generate an error.

Channel Number

The channel number identifies what channel will be switched to its COM terminal. Channel numbers are 0 to 7. Only valid channels can be accessed in a channel list. When switching the control relays, the channel number (0 to 6) identifies which control relay will be switched (see Figure 1-1).

Examples: Multiplexer Module Channel List

One-wire operating mode:

CLOSe (@10173)	*!Connect card 01, bank 7, channel 3 HI terminal to the one-wire HI COM terminal*

Two-wire operating mode:

CLOSe (@173,176)	*!Connect card 01, bank 7, channels 3 and 6 HI and LO terminals, to bank 7 HI and LO COM terminals*

Three-wire operating mode:

CLOSe(@133:136)	*!Connect card 01, bank 3, channels 3 through 6 HI and LO terminals, to bank 3 HI and LO COM terminals. Also connect bank 7, channels 3 through 6 LO terminal, to bank 7 LO COM terminal.*

Four-wire operating mode:

CLOSe(@133:136,233:236)	*!Connect cards 01 and 02, bank 3, channels 3 through 6 HI and LO terminals, to bank 3 HI and LO COM terminals. Also, connect bank 7, channels 3 through 6 HI and LO terminals, to bank 7 HI and LO COM terminals.*

Control relays:

CLOSe (@10995)	*!Connect the upper and lower 32 channels together for a 64-channel two-wire*

Initial Operation

You can use the following example program to verify initial multiplexer module operation by closing a channel and querying channel closure. The example first resets the switchbox and then closes bank 0, channel 2 of a single multiplexer module (card number 1) in the switchbox.

The program next queries the channel closure state. A returned "1" shows that the command to close the channel has been sent to the switchbox. A returned "0" shows that the command to close the channel has not been sent to the switchbox.

BASIC is used as the program language. The computer interfaces to the mainframe using GPIB with interface select code 7, primary address 09, and secondary address 14. This example rsesets the switchbox and closes card 01 bank 0 channel 2 (to COM).

```
10  OUTPUT 70914;"RST"                  !Reset the module, set all relays to open
20  OUTPUT 70914;"CLOS(@102)"           !Connect bank 0 channel 2 HI and LO
                                         terminals to bank 0 to COM HI and LO
                                         terminals
30  OUTPUT 70914;"CLOS? (@102)"         !Query channel 02
40  ENTER 70914;Value                   !Enter results into Value
50  PRINT Value                         !Display results (should return "1")
60  END
```

Chapter 2
Using the Relay Multiplexer

Using This Chapter

This chapter shows how to use the Relay Multiplexer module, including:

- Multiplexer Commands/States 35
- Switching Channels 37
- Scanning Channels 43
- Miscellaneous Multiplexer Functions 51

Multiplexer Commands/States

This section summarizes Relay Multiplexer module commands, queries, and reset states. Table 2-1 shows multiplexer commands used in this chapter. See Chapter 3 for additional information about the commands.

Table 2-1. Selected Multiplexer Commands Used in Chapter 2

Command	Description
INITiate[:IMMediate]	Starts the scan sequence and closes the first channel in the *channel_list*.
OUTPut:TTLTrg*n*[:STATe] ON	Enables selected output to trigger pulses from the TTL Trigger bus *line* specified.
OUTPut:EXTernal][:STATe] ON	Enables selected output to trigger pulses from command module's "Trig Out" port.
[ROUTe:]CLOSe <*channel_list*>	Closes the channels in the *channel_list*.
[ROUTe:]CLOSe? <*channel_list*>	Queries the state of the closed channels in the *channel_list*.
[ROUTe:]FUNCtion <*card_number*>,<*function*>	Sets the operating mode to one-wire, two-wire, three-wire, or four-wire.
[ROUTe:]OPEN <*channel_list*>	Opens the channels in the *channel_list*.
[ROUTe:]SCAN <*channel_list*><	Defines the channels to be scanned. Channels specified in the *channel_list* are closed one at a time.
[ROUTe:]SCAN:PORT	Closes bank COM terminals to the analog bus during a scan.
[ROUTe:]SCAN:MODE	Sets the scan mode to volts, 2-wire ohms, or 4-wire ohms.
TRIGger:SOURce <*source*>	Selects the trigger *source* to advance the scan.
*CLS	Clears all switchbox status registers and error queue.
*ESE	Enables event status register.
*RST	Resets the hardware and software to a known state.
*SRE	Enables status register.

Table 2-2 summarizes the query commands you can use to determine the configuration or state of the multiplexer. All commands put the data into the output buffer where you can retrieve it to your computer.

Table 2-2. Multiplexer Query Commands

Command	Description	Command	Description
ARM:COUN?	Number of Scanning Cycles	SCAN:PORT?	Scanning Port Selected
CLOS?	Channel Closed	STAT:OPER:ENAB?	Status Operation Enable
FUNC?	Operating Mode Selected	STAT:OPER:EVEN?	Status Operation Event
OPEN?	Channel Open	SYST:CDES? <*number*>	Module Description
INIT:CONT?	Scanning State	SYST:CTYP? <*number*>	Module Type
OUTP:ECLTrg*n*?	ECL Trigger Output State	SYST:ERR?	System Error
OUTP:EXT?	External Trigger Output State	TRIG:SLOP?	Trigger Slope
OUTP:TTLTrg*n*?	TTL Trigger Output State	TRIG:SOUR?	Trigger Source
SCAN:MODE?	Scanning Mode Selected		

Table 2-3 lists the parameters and default values for the functions described in this chapter. When the multiplexer is switched on or *RST (reset), all bank channels are set to open and the current *channel_list* for scanning is invalidated.

Table 2-3. Multiplexer Reset Conditions

Parameter	Default	Description
ARM:COUNt	1	Number of scanning cycles is one
INITiate:CONTinuous	OFF	Number of scanning cycles is set by ARM:COUNt
OUTPut:ECLTrg*n*[:STATe]	OFF	Trigger output from ECLT sources is disabled
OUTPut[:EXTernal][:STATe]	OFF	Trigger output from external sources is disabled
OUTPut:TTLTrg*n*[:STATe]	OFF	Trigger output from TTLT sources is disabled
[ROUTe:]SCAN:MODE	NONE	Channel list volts/ohms measurements disabled
[ROUTe:]SCAN:PORT	NONE	Analog bus port connection disabled
TRIGger:SOURce	IMM	Will advance scanning cycles automatically

Switching Channels

For general purpose switching, you can switch channels (connect or disconnect signals) in one-wire, two-wire, three-wire, or four-wire operating modes by opening or closing specific channel(s).

NOTE *For more information, see the [ROUTe:]FUNCtion command. There is no need to send the [ROUTe:]FUNCtion command if the status register switch (see "Setting the Status Register Switch") is set to the correct operating mode.*

Switching Channels Comments

Setting Multiplexer Function. Use FUNCtion <*card_number*>,<*function*> to configure the Relay Multiplexer, where <*function*> = WIRE1 | WIRE2 | WIRE2X64 | WIRE3 | WIRE4.

Opening/Closing Channels. Use CLOSe <*channel_list*> to close bank channel(s) and OPEN <*channel_list*> to open bank channel(s). *Channel_list* has the form (@*ss0hbc*) where *ss* = card number (00-99), *0h* = one-wire mode only HI/LO switching (00 or 01), *b* = bank number (0-7), and *c* = channel number (0-7).

Opening/Closing Multiple Channels. To close or open multiple channels, place a comma (**,**) between the channel numbers. To close or open a range of channels, place a colon (**:**) between the channel numbers. You can do this for both single or multiple module switchboxes. See [ROUTe:]OPEN and [ROUTe:]CLOSe for additional information.

Querying Open/Closed Channels. The CLOS? <*channel_list*> and OPEN? <*channel_list*> commands determine if the channel in the *channel_list* is open or closed, respectively. (The query command does not determine if, in the event of a hardware failure, the channel remains open/closed.) See [ROUTe:]OPEN? and [ROUTe:]CLOSe? for additional information.

Switching Control Relays. The control relays 0990 to 0996 can also be switched using the OPEN and CLOSe commands, provided the FUNCtion command is executed first. See [ROUTe:]OPEN and [ROUTe:]CLOSe for additional information.

FRES: When operating in one-wire mode, 4-wire resistance measurement (FRES) is not supported. See the [ROUTe:]SCAN:MODE command for additional information.

Analog Bus Connection When Scanning. In all four modes of operation, the analog bus can be connected during a scan using the SCAN:PORT command. In three-wire mode, the paired bank (4-7) channel LO terminal can be connected to the analog bus Guard terminal. See the [ROUTe:]SCAN:PORT command for additional information.

Analog Bus Connection When Not Scanning. When opening and closing individual channels in all four modes of operation, the analog bus can be connected by switching the control relays (0992-0994, 0996) using the OPEN and CLOSe commands. See [ROUTe:]OPEN and [ROUTe:]CLOSe for additional information.

Relay Switch Card Configuration. In all modes of operation the relay switch card wire jumpers can be changed to 1x8 or 1x16 configurations as required. See “Configuring the Switch Card Wire Jumpers” for additional information.

Switching Channels Examples

Four example programs follow that illustrate one-wire, two-wire, three-wire, and four-wire modes of operation for switching multiplexer channels. The examples are:

- Example: Switching Channels (One-Wire)
- Example: Switching Channels (Two-Wire)
- Example: Switching Channels (Three-Wire)
- Example: Switching Channels (Four-Wire)

Example: Switching Channels (One-Wire)

This example illustrates one-wire mode operation. For the example, the HI terminal is used. Bank 2 channel 1 is closed, connecting the HI terminal to the one-wire HI COM terminal. Figure 2-1 shows how the multiplexer is configured.

For one-wire operation, the control relays are set as follows. 0990 depends on HI or LO terminal selection. 0991/0995 are set closed. 0992 will close when SCAN:PORT ABUS is selected during a scan (see “Scanning Channels”). 0993/0994/0996 remain in current state (open if not changed after *RST)

To connect the HI terminals of bank 2 channel 1 to the one-wire HI COM terminal, execute:

```
FUNC 1,WIRE1          !Configures the multiplexer (card 01) for
                       one-wire operation

CLOS (@10121)         !Connects the HI terminal of bank 2 channel
                       1 to the one-wire HI COM terminal
```

NOTE *If the status register switch is set to one-wire operating mode, the FUNC 1,WIRE1 command is not required. When operating in the one-wire mode, only one channel at a time can be closed.*

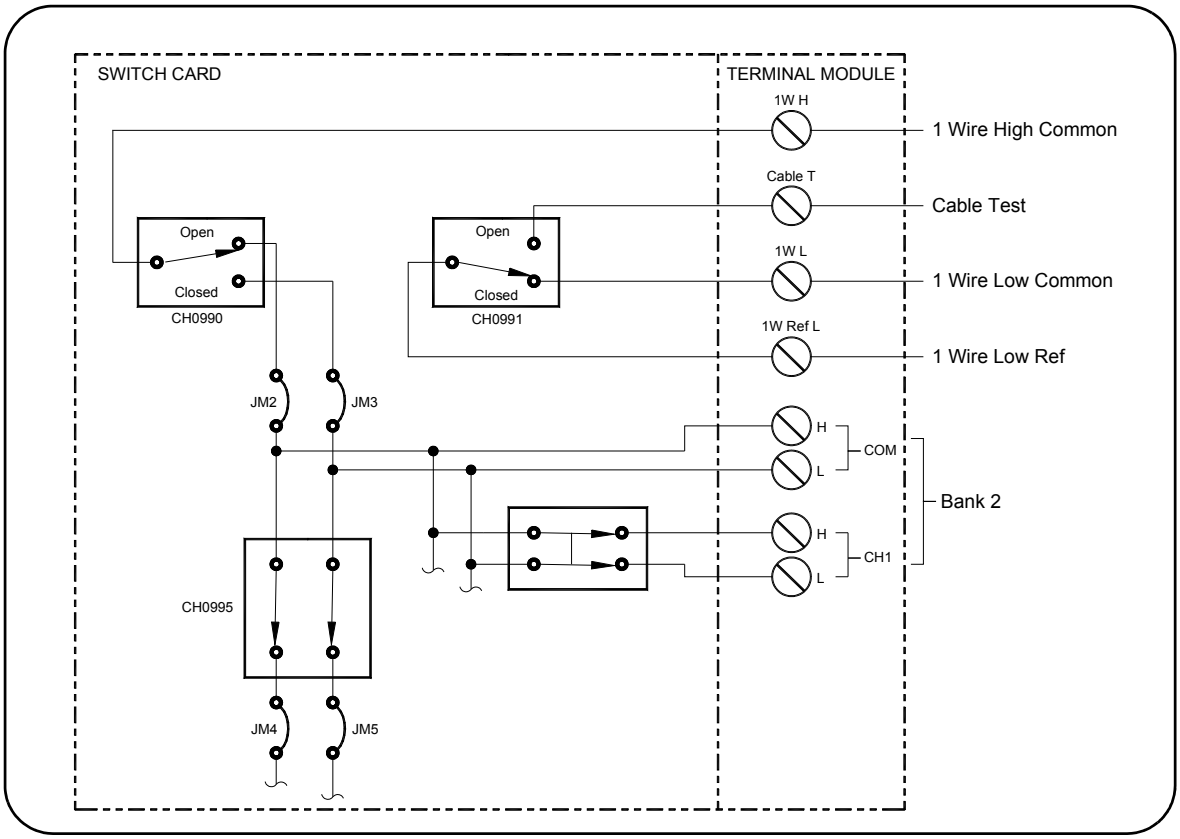


Figure 2-1. Example: Switching Channels (One-Wire)

Example: Switching Channels (Two-Wire)

This example illustrates two-wire mode operation. The HI and LO terminals of bank 0 channels 0 and 7 are closed, connecting them to the bank 0 HI and LO COM terminals. Figure 2-2 shows how the multiplexer is configured.

For two-wire operation, the control relays are set as follows. 0990/0991 are opened if using the <*channel_list*> command with SCAN:PORT ABUS and SCAN:MODE <*mode*>. *Mode* can be RES, VOLT, or NONE. 0990/0991 are left in their present state if *mode* is FRES. 0992/0993 will close when SCAN:PORT ABUS is selected during a scan (see “Scanning Channels”).

0994/0995/0996 remain in their present state with the following exceptions. 0994 is closed in RES mode. If <*card_number*>, WIRE2X64 (2x64 configuration), 0994 is closed in the RES and NONE modes. In the FRES mode, 0994 and 0995 are opened. 0996 closes and connects COM to LO for voltage measurements with the MEASure or SCPI commands in a scanning multimeter.

To connect the HI and LO terminals of bank 0 channels 0 and 7 to the bank 0 COM terminals, execute:

```
FUNC 1,WIRE2          !Configures the multiplexer (card #1) for
                       two-wire operation

CLOS @100,107)        !Connects the HI and LO terminals of bank 0
                       channels 0 and 7 to bank 0 COM terminals
```

NOTE *If the Status Register switch is set to the two-wire operating mode, the FUNC 1,WIRE2 command is not required. The WIRE2X64 command can be used rather than closing control relay 0995 to configure the card to a single 64-channel multiplexer. (Available only with E1406A (Switchbox rev. A.06.00 or later)).*

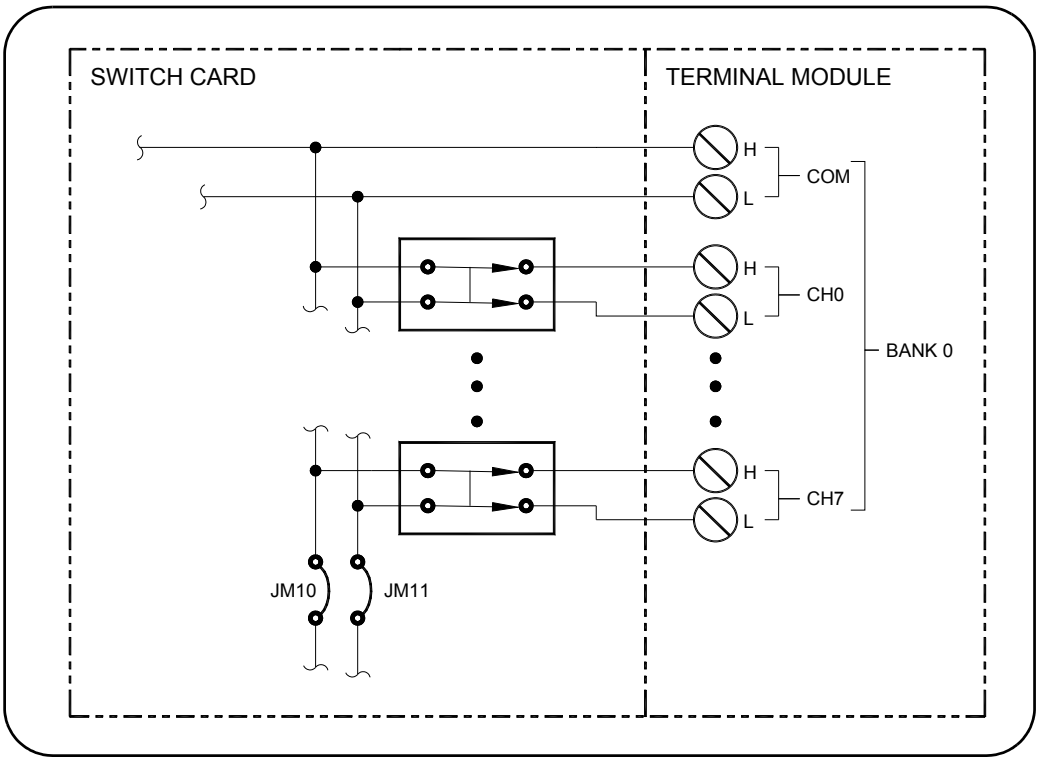


Figure 2-2. Example: Switching Channels (Two-Wire)

Example: Switching Channels (Three-Wire)

This example illustrates three-wire mode operation. The HI and LO terminals of bank 0 channel 0 are closed, connecting them to the bank 0 COM terminals. The LO terminal of bank 4 channel 0 is closed, connecting it to the bank 4 LO COM terminal. Figure 2-3 shows how the multiplexer is configured.

For three-wire operation, the control relays are set as follows. 0990/0991 are set open when <*channel_list*> is executed. 0992/0993/0996 will close when SCAN:PORT ABUS is selected during a scan. 0992 and 0993 are opened when not SCAN:PORT ABUS (see “Scanning Channels”). 0994/0995 are set open when SCAN <*channel_list*> is executed

To connect the HI and LO terminals of bank 0 channel 0 and the LO terminal of bank 4 channel 0 to their COM terminals, execute:

```
FUNC 1,WIRE3        !Configures the multiplexer (card 01) for
                     three-wire operation

CLOS (@100)         !Connects the HI and LO terminals of bank 0
                     channel 0 to the bank 0 COM terminals and
                     the LO terminal of bank 4, channel 0 to the
                     bank 4 LO COM terminal
```

NOTE *If the Status Register switch is set to three-wire operating mode, the FUNC 1,WIRE3 command is not required. In three-wire mode, banks are paired 0/4, 1/5, 2/6, and 3/7. Do not connect user wiring to the HI terminal in the upper bank pair (4-7), as this terminal is switched during three-wire operation. Upper bank pair (4-7) channels cannot be switched or queried while in this mode.*

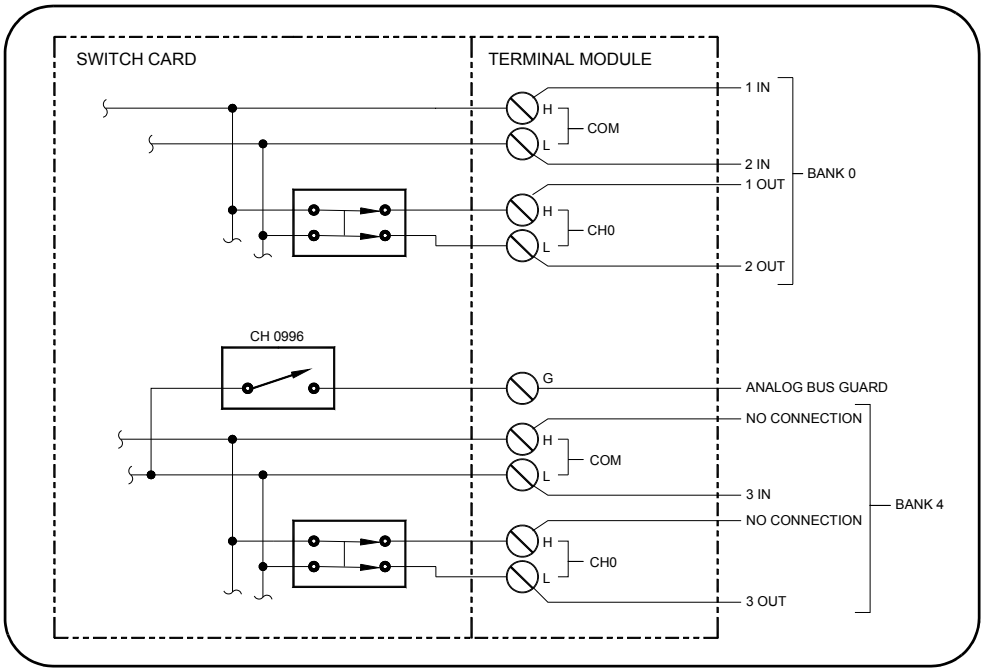


Figure 2-3. Example: Three-Wire Mode Channel Switching

Example: Switching Channels (Four-Wire)

This example illustrates four-wire mode operation. The HI and LO terminals of bank 0 channel 0 are closed, connecting them to the bank 0 COM terminals. At the same time, the HI and LO terminals of bank 4 channel 0 are closed, connecting them to the bank 4 COM terminals. Figure 2-4 shows how the multiplexer is configured.

For four-wire operation, the control relays are set as follows. 0990/0991 are set open when SCAN <*channel_list*> is executed. 0992/0993 will close when SCAN:PORT ABUS is selected during a scan. They are opened otherwise (see “Scanning Channels”). 0994/0995/0996 are set open when SCAN <*channel_list*> is executed.

To connect the HI and LO terminals of bank 0 channel 0 and the HI and LO terminals of bank 4 channel 0 to their COM terminals, execute:

```
FUNC 1,WIRE4        !Configures the multiplexer (card #1) for
                     four-wire operation.

CLOS (@100)         !Connects the HI and LO terminals of bank
                     4 channel 0 to the bank 4 COM terminals
```

NOTE *If the Status Register switch is set to four-wire operating mode, the FUNC 1,WIRE4 command is not required. In four-wire mode, banks are paired 0/4, 1/5, 2/6, and 3/7. Upper bank pair (4-7) channels cannot be switched or queried while in this mode.*

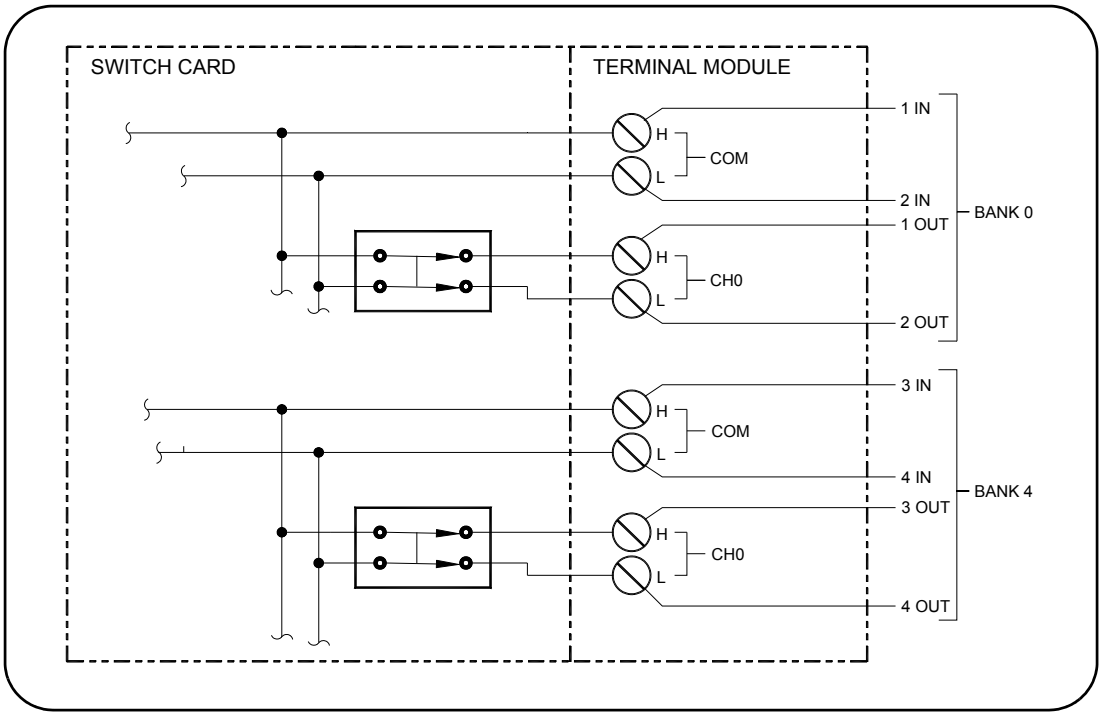


Figure 2-4. Example:Four-Wire Mode Channel Switching

Scanning Channels

Scanning the multiplexer module channels consists of closing bank channel(s) to the respective bank COM terminal(s) one channel at a time. Single scan, multiple scans (2 to 32767), or continuous scanning modes are available.

Scanning Channels Comments

Scanning Channels Sequence. The TRIGger:SOURce command specifies the source to advance the scan. The OUTPut command can be used to enable the E1406A Command Module "Trig Out" port, TTL Trigger bus lines (0-7), or ECL Trigger bus lines (0-1). Figure 2-5 illustrates scanning.

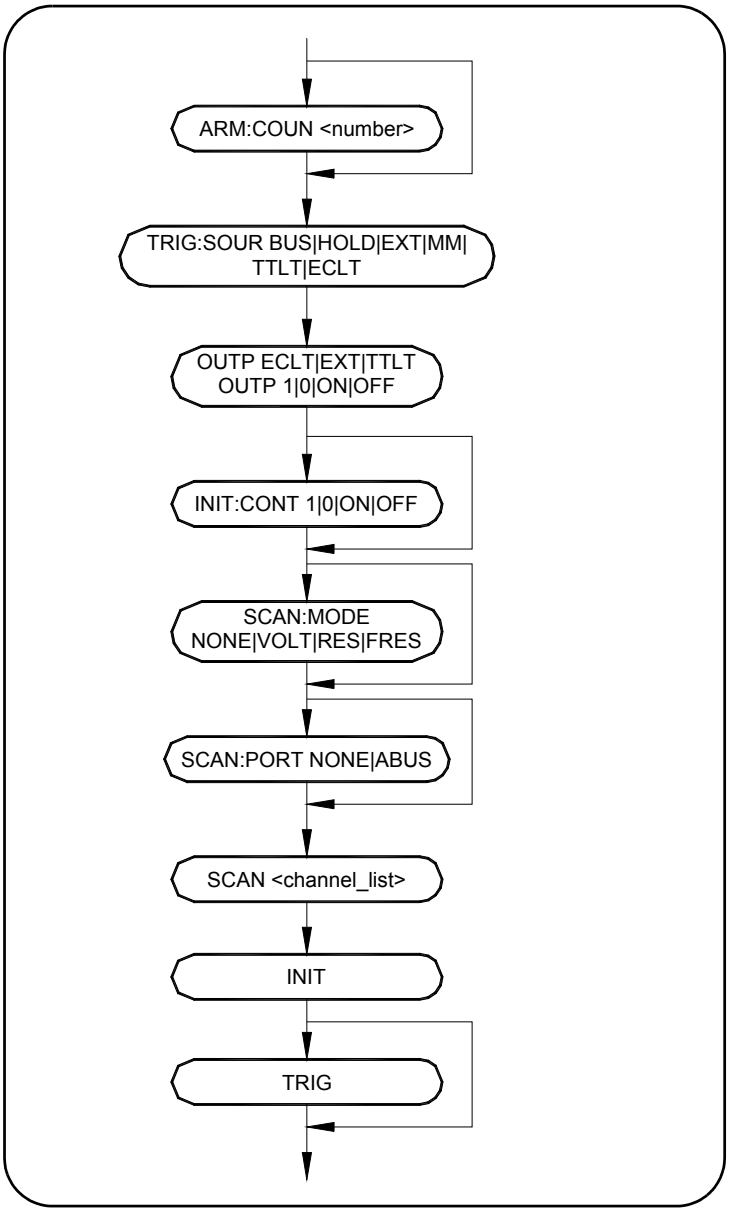


Figure 2-5. Scanning Channels Sequence

Scanning Requirements of a Switchbox (With a Command Module). To scan modules in a switchbox, you must nnow the card numbers of all the modules to be scanned, sequentially address the modules (for example, logical address 112, 113, 114, etc.), and set the lowest addressed module to a logical address that is a multiple of 8.

Channel List Can Be Extended Across Boundaries. For multiple-module switchbox instruments, the channels to be scanned can extend across switch modules. For example, for a two-module switchbox instrument, SCAN (@100:277) will scan all channels of both multiplexer modules.

Setting Multiple Continuous Scans. Use ARM:COUNt <*number*> to set from 1 to 32767 scans. Use INITiate:CONTinuous ON to set continuous scanning.

Control Relay Switching. Control relays (0990 to 0996) are not affected by opening and closing of the channel relays (banks 0 to 7). They are switched when configuring a mode (See "Channel Switching" in this chapter), and during scanning when SCAN:PORT ABUS is selected (see the [ROUTe:]SCAN:PORT command for more information).

Two-Wire Ohms Measurements. When making two-wire ohms scanning measurements using multimeters with SOURCE/SENSE leads, use SCAN:MODE RES. When making two-wire ohms scanning measurements using multimeters with HI/LO leads, use SCAN:MODE VOLT. See the [ROUTe:]SCAN:MODE command for more information.

Scanning Channels Examples

Some example programs follow that show how to scan channels using the E1406A Command Module and/or external multimeters. The examples are:

- Example: Scanning Channels Using E1406A Command Module
- Example: Scanning Channels Using E1412A Multimeter
- Example: Scanning Channels Using 3457A Multimeter
- Example: Scanning Multimeter DCV Measurements
- Example: Scanning Multimeter Resistance Measurements

Example: Scanning Channels Using E1406A Command Module

This example uses the E1406A Command Module TTL Trigger Bus lines to synchronize E1460A multiplexer bank 0 channel 0, to bank 7 channel 7 closures to an E1411B System Multimeter. For the example, a two-wire ohms measurement is performed.

For measurement synchronization, E1406A TTL Trigger Bus line 0 is used by the multiplexer to trigger the multimeter to perform a measurement. The E1406A TTL Trigger Bus line 1 is used by the multimeter to advance the multiplexer scan.

Figure 2-6 shows how to connect the E1460A multiplexer module to the E1411B multimeter. This example uses GPIB select code 7, primary address 09, and secondary address 03 for the multimeter and GPIB select code 7, primary address 09, and secondary address 14 for the multiplexer.

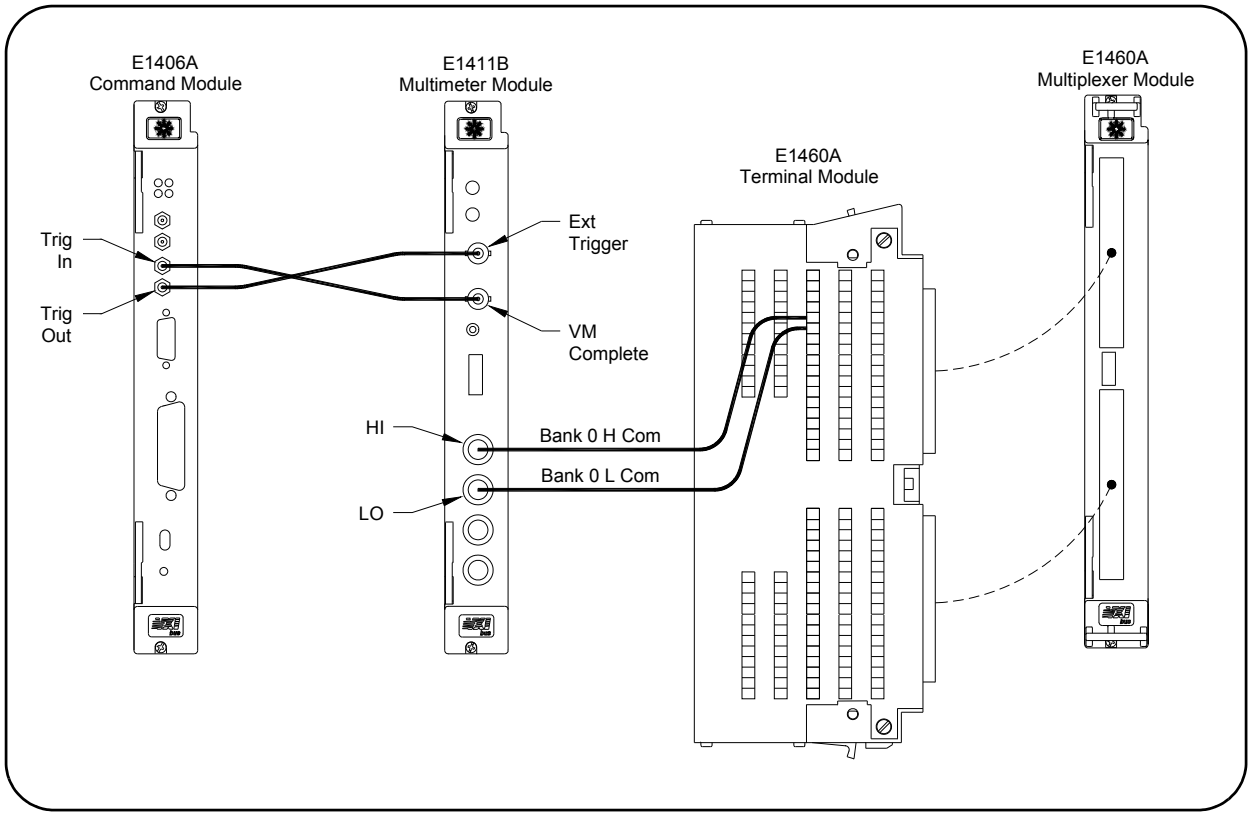


Figure 2-6. Example: Scanning Channels Using E1406A Command Module

```
10  DIM Readings(1:64)                    !Dimensions computer to store readings
20  OUTPUT 70903;"*RST"                   !Resets multimeter module to known state
30  OUTPUT 70903;"CONF:RES AUTO,MAX"      !Configures the multimeter to measure
                                           resistance using autorange at the least
                                           accurate resolution (the largest value)
40  OUTPUT 70903;"TRIG:SOUR TTLT0"        !Multimeter to perform measurement when
                                           trigger received on TTL Trigger bus line 0
50  OUTPUT 70903;"OUTP:TTLT1:STAT 1"      !Multimeter to cause trigger on TTL trigger bus
                                           line 1 when measurement complete
60  OUTPUT 70903;"TRIG:COUN 64"           !Multimeter to receive 64 triggers
70  OUTPUT 70903;"*OPC?"                  !Multimeter operations complete
80  ENTER 70903;Opc                       !Enters a "1" when complete
90  OUTPUT 70914;"*RST"                   !Resets the multiplexer to known state
100 OUTPUT 70914;"FUNC 1,WIRE2X64"        !Configures multiplexer for 64 channels and
                                           closes control relay 0995
110 OUTPUT 70914;"OUTP:TTLT0:STAT 1"      !Multiplexer to cause trigger on TTL trigger bus
                                           line 0 when channel close complete
```

```
120 OUTPUT 70914;"TRIG:SOUR TTLT1"      !Multiplexer to advance scan when trigger
                                         received in TTL trigger bus line 1
130 OUTPUT 70914;"SCAN:MODE VOLT"       !Sets switchbox measurement to volt (used
                                         to make 2-wire resistance measurement on
                                         multimeter's HI/LO terminals)
140 OUTPUT 70914;"SCAN:PORT ABUS"       !Closes control relays 992 and 993 connecting
                                         the analog bus to the upper and lower four
                                         bank commons
150 OUTPUT 70914;"SCAN (@100:177)"      !Defines channel list to scan bank 0 channel
                                         0 to bank 7 channel 7
160 OUTPUT 70914;"*OPC?"                !Multiplexer operations complete
170 ENTER 70914;Opc                     !Enters a "1" when complete
180 OUTPUT 70903;"READ?"                !Places multimeter in wait-for-trigger state.
                                         Will send measurement results to output
                                         buffer when triggered.
190 OUTPUT 70914;"INIT"                 !Closes bank 0 channel 0 and enables the
                                         scan. Causes a trigger output on TTL Trigger
                                         bus line 0 that initiates the multimeter to
                                         make a measurement.
200 ENTER 70903;Readings(*)             !Enters measurement results
210 PRINT Readings(*)                   !Displays measurement result
220 END                                 !Terminates program
```

Example: Scanning Channels Using E1412A Multimeter

This example program uses the E1406A Command Module for one-wire scanning of the E1460A using the two-wire ohms function of the E1412A multimeter. The TTL trigger bus lines perform E1412A multimeter triggering and E1460A multiplexer channel advance. This program uses the stand-alone switchbox mode.

For this example, the following resistors are connected to the channels indicated and to the one-wire COM terminal. The remaining channels are open.

1 kΩ on channel 00 HI
1.5 kΩ on channel 00 LO
1.2 kΩ on channel 01 HI
1.8 kΩ on channel 01 LO

A typical result returns +0, "No error" for the response to the SYST:ERR? command. The four channels with resistors connected return the nominal value of the resistor, such as 1003.129 for channel 00 HI or 1489.102 for channel 00 LO. All other channels return 9.9E+37 to indicate an open channel.

```
10   ASSIGN @Dvm TO 70903
20   ASSIGN @Mux TO 70914
30   DIM A$[80],Rdgs(1:64)
40   CLEAR @Dvm
50   CLEAR @Mux
60   OUTPUT @Dvm;"*RST;*CLS"
70   OUTPUT @Mux;"*RST;*CLS"
80   OUTPUT @Dvm;"FUNC:RES"
90   OUTPUT @Dvm;"TRIG:SOUR TTLT0"
100  OUTPUT @Dvm;"TRIG:COUN 64"
110  OUTPUT @Dvm;"OUTP:TTLT1:STAT ON"
120  OUTPUT @Dvm;"*OPC?"
130  ENTER @Dvm;Cp
140  OUTPUT @Dvm;"SYST:ERR?"
150  ENTER @Dvm;A$
160  PRINT A$
170  OUTPUT @Mux;"FUNC 1,WIRE1"
180  OUTPUT @Mux;"OUTP:TTLT0:STAT 1"
190  OUTPUT @Mux;"TRIG:SOUR TTLT1"
200  OUTPUT @Mux;"SCAN:MODE RES"
210  OUTPUT @Mux;"SCAN:PORT ABUS"
220  OUTPUT @Mux;"SCAN (@100:177)"
230  OUTPUT @Mux;"*OPC?"
240  ENTER @Mux;Cp
250  OUTPUT @Mux;"SYST:ERR?"
260  ENTER @Mux;A$
270  PRINT A$
280  OUTPUT @Dvm;"INIT"
290  OUTPUT @Mux;"INIT"
300  OUTPUT @Dvm;"FETCh?"
310  ENTER @Dvm;Rdgs(*)
320  PRINT Rdgs(*)
330  END
```

Example: Scanning Channels Using 3457A Multimeter

This example uses the E1406A Command Module "Trig In" and "Trig Out" ports to synchronize the multiplexer module bank 0 channel 0 to 2 closures to an external 3457A multimeter. The multimeter's reading storage capacity is used to store measurement results.

For measurement synchronization, the E1406A "Trig Out" port is connected to the 3457A multimeter "Ext Trig" port. This trigger signals the multimeter to make the measurement. The E1406A "Trig Out" port is connected to the 3457A multimeter "Voltmeter Complete" port. This trigger causes the multiplexer to advance the scan.

Figure 2-7 shows how to connect the command module and multiplexer module to the 3457A multimeter. This example uses GPIB select code 7, primary address 09, and secondary address 14 for the multiplexer and GPIB select code 7 and primary address 22 for the 3457A digital multimeter.

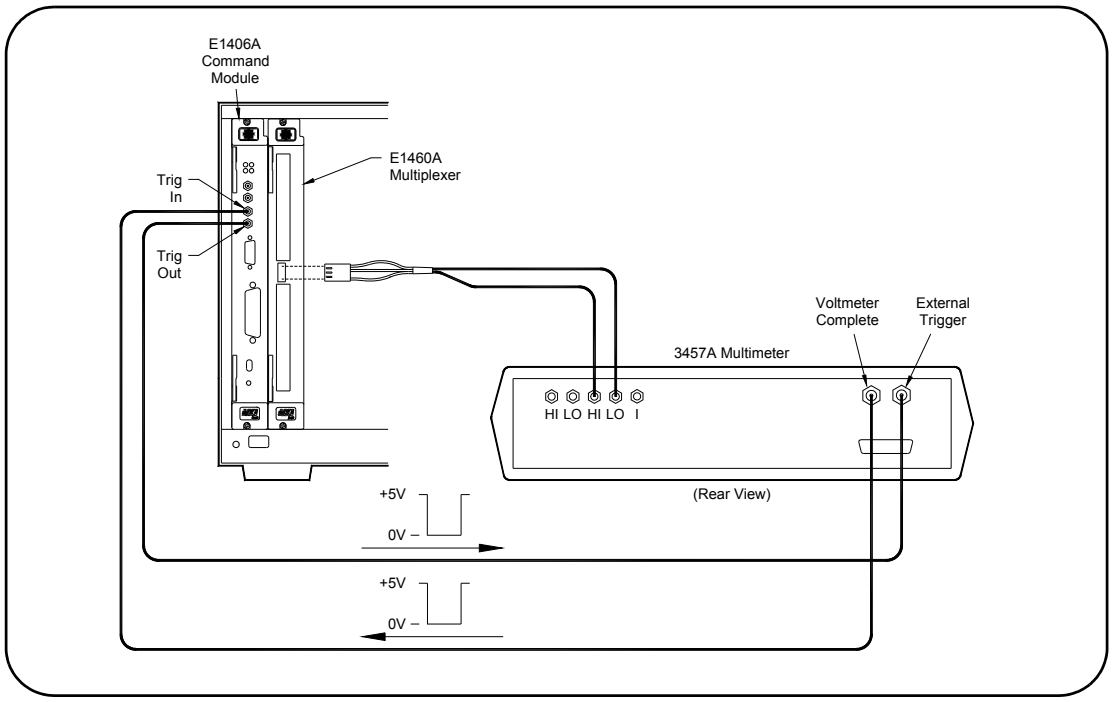


Figure 2-7. Example: Scanning Channels Using 3457A Multimeter

```
10  OUTPUT 722;"TRIG EXT;DCV;MEM FIFO"      !Configures the 3457A multimeter to external
                                              trigger to measure DCV and store readings
20  OUTPUT 70914;"*RST;*CLS"                !Resets multiplexer module to a known state
30  OUTPUT 70914;"OUTP ON"                  !Enables E1406A "Trig Out" port
40  OUTPUT 70914;"TRIG:SOUR EXT"            !Sets switchbox trigger source to external
                                              triggering
50  OUTPUT 70914;"SCAN:MODE VOLT"           !Sets switchbox measurement mode
60  OUTPUT 70914;"SCAN (@10000:10015)"      !Defines channel list
70  OUTPUT 70914;"INIT"                     !Closes bank 0 channel 0 and enables the
                                              scan. Causes a trigger output from E1406A
                                              "Trig Out" port which initiates the external
                                              multimeter to make a measurement. When the
                                              measurement is complete, the multimeter's
                                              "VM Complete" port sends a trigger to the
                                              multiplexer to advance the scan.
80   Wait 1                                 !Wait 1 second
90   FOR Channels = 1 to 16                 !Start counting loop (16 channels)
100      ENTER 722;Results                  !Enter measurement result
110  NEXT Channels                          !Increment count and repeat measurement
                                              process for 16 measurements
120  END                                    !Terminate program
```

Example: Scanning Multimeter DCV Measurements

This example uses the E1406A Command Module, E1411B System Multimeter, and E1460A multiplexer modules to perform a DC voltage measurement on all 64 channels in a scanning multimeter configuration. In the scanning multimeter configuration, the multiplexer module's logical address must be set one number higher than the multimeter module.

Figure 2-8 shows how to connect the multiplexer module to the multimeter module. This example uses GPIB select code 7, primary address 09, and secondary address 03 for the scanning multimeter (multimeter/multiplexer).

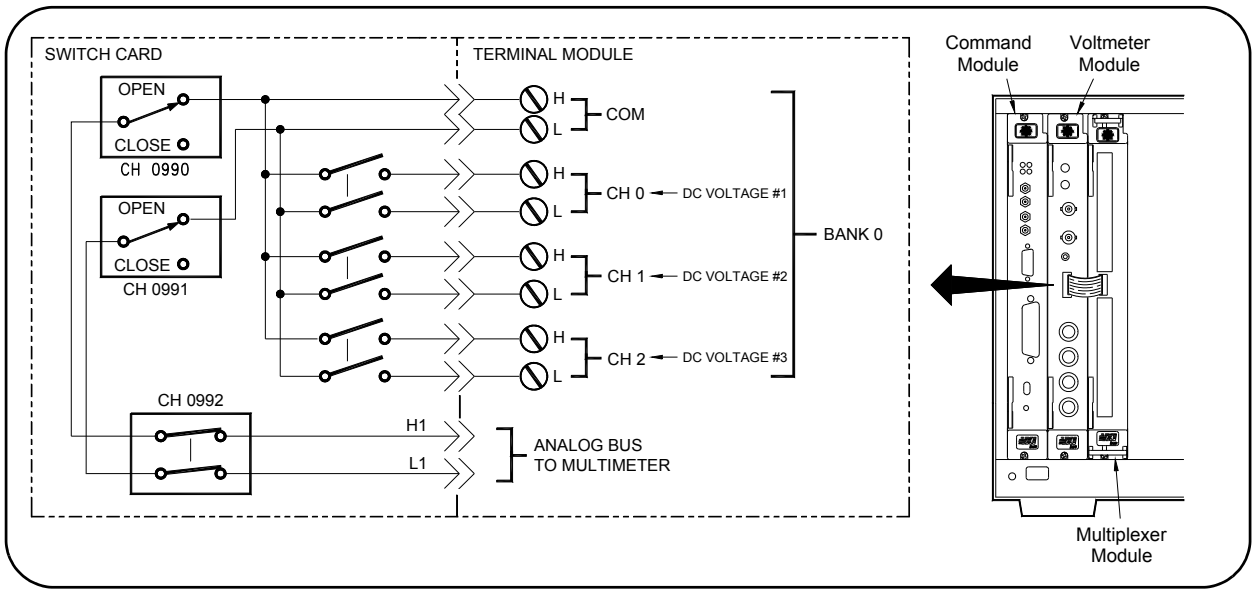


Figure 2-8. Example: Scanning Multimeter DCV Measurements

```
10  DIM Rdgs(1:64)                                  !Dimension computer to store readings
20  OUTPUT 70903;"*RST"                             !Reset the scanning multimeter module to a
                                                     known state
30  OUTPUT 70903;"ROUT:FUNC1,WIRE2X64"              !Required to close control relay 0995 to access
                                                     upper 4 banks
40  OUTPUT 70903;"MEAS:VOLT:DC? (@100:177)"         !Configure the scanning multimeter to measure
                                                     voltage on bank 0 channels 0 to 2
50  ENTER 70903;Rdgs(*)                             !Enter measurement result
60  FOR I = 1 TO 64                                 !Start counting loop (3 channels)
70      PRINT Rdgs(I)                               !Display measurement result
80  NEXT I                                          !Increment the count and repeat measurement
                                                     process for a total of 3 measurements
90  END                                             !Terminate program
```

Example: Scanning Multimeter Resistance Measurements

This program uses an E1406A Command Module to verify the E1460A multiplexer will work in one-wire mode for resistance measurements with the E1411B multimeter when the two are instruments are configured as a scanning multimeter.

For this example to work in either switchbox or scanning multimeter mode, you must externally connect the HI current source banana jack to the HI input banana jack and the LO current Source/COM to the LO input banana jack on the front panel of the E1411B multimeter.

For this example, the following resistors are connected to the one-wire COM terminal. The remaining channels are open.

1 kΩ on channel 00 HI
1.5 kΩ on channel 00 LO
1.2 kΩ on channel 01 HI
1.8 kΩ on channel 01 LO

A typical result returns HEWLETT-PACKARD,E1411B,0,A.04.02 in response to the *IDN? command. Also, "If E1460A terminal module jumper 10 is cut and jumpers 11, 12, and 13 are in place, ROUT:FUNC? will return "WIRE1"" is displayed.

The four channels with resistors connected return the nominal value of the resistor, such as 1003.129 for channel 00 HI or 1489.102 for channel 00 LO. All other channels return 9.9E+37 to indicate an open channel.

```
10   DIM A$[40],Rdgs(1:66)
20   ASSIGN @Dvm TO 70903
30   OUTPUT @Dvm;"*RST;*CLS"
40   WAIT 2
50   OUTPUT @Dvm;"*IDN?"
60   ENTER @Dvm;A$
70   PRINT A$
80   OUTPUT @Dvm;"ROUT:FUNC? 1"
90   ENTER @Dvm;A$
100  PRINT "If E1460A terminal module jumper 10 is cut, and jumpers 11"
110  PRINT "12, and 13 are in place, ROUT:FUNC? will return 'WIRE1'."
120  PRINT A$
130  OUTPUT @Dvm;"MEAS:RES? (@10000:10101)"
140  ENTER @Dvm;Rdgs(*)
150  PRINT Rdgs(*)
160  END
```

Miscellaneous Multiplexer Functions

This section describes some miscellaneous multiplexer functions, including:

- Using the Scan Complete Bit
- Using the Analog Bus
- Saving and Recalling States
- Detecting Error Conditions
- Synchronizing the Multiplexer

Using the Scan Complete Bit

The scan complete bit (bit 8) can be used in the Operation Status Register of a switchbox to determine when a scanning cycle completes (no other bits in the register apply to the switchbox). Bit 8 has a decimal value of 256 and you can read it directly with the STAT:OPER? command.

When enabled by the STAT:OPER:ENAB 256 command, the scan complete bit will be reported as bit 7 of the Status Byte Register. Use the GPIB Serial Poll or the IEEE 488.2 Common Command *STB? to read the Status Byte Register. When bit 7 of the Status Byte Register is enabled by the *SRE 128 common command to assert a GPIB Service Request, you can interrupt the computer when the scan complete bit is set, after the scanning cycle completes. This allows the controller to do other operations while the scanning cycle is in progress.

Example: Using the Scan Complete Bit

This example monitors bit 7 in the Status Byte Register to determine when the scanning cycle completes. The example uses GPIB select code 7, primary address 09, and secondary address 14 for the multiplexer.

```
10   OUTPUT 70914;"*CLS"                    !Clear all switchbox status structure
20   OUTPUT 70914;"STAT:OPER:ENAB 256"      !Enable scan complete bit to set bit 7 in
                                             Status Byte Register
30   OUTPUT 70914;"*SRE 128"                !Enable bit 7 of Status Byte Register to
                                             assert RQS
40   OUTPUT 70914;"TRIG:SOUR EXT"           !Set to external trigger mode
50   OUTPUT 70914;"SCAN (@100:102)"         !Defines channel list to scan bank 0 channels
                                             0 through 2
60   OUTPUT 70914;"INIT"                    !Start scanning cycle
70   WHILE NOT BIT(SPOLL(70914),7)          !Waiting for scan complete
80       PRINT "DO OTHER OPERATION HERE"    !Enter program lines for computer to do other
                                             operations
90   END WHILE
100  PRINT "INTERRUPT GENERATED"            !Program goes to this line after interrupt is
                                             generated by a completed scanning cycle
110  END                                    !Terminate program
```

Using the Analog Bus

The multiplexer can be configured to perform voltage, two-wire ohm, or four-wire ohm measurements using the analog bus. These measurements can be performed by switching or scanning channels (refer to the previous examples).

By switching the control relays (0990 to 0996), the COM lines can be connected to the analog bus connection for measurement using a VXI multimeter (such as the E1411B) or external multimeter (such as the 3457A). A special terminal enables the multiplexer to perform cable or harness testing. You can connect the multiplexer analog bus to other measurement or switching devices to perform measurements.

During a scan, the control relays (0992 to 0994 and 0996) are automatically switched when configured using the SCAN:PORT command. See [ROUTe:]SCAN:PORT on for more information.

When switching channels using the OPEN and CLOSe commands, the analog bus must be manually connected. When the FUNCtion command is executed, all analog bus control relays are opened. If an analog bus connection is required, you must close the appropriate control relay. Once switched, the relay remains closed unless specifically opened (OPEN command, power-up, or *RST). Control relay numbers and functions follow.

- 0992: Connects lower 32 channels (banks 0 to 3) to the analog bus H1 terminals.
- 0993: Connects upper 32 channels (banks 4 to 7) to the analog bus H2 terminals.
- 0994: Connects lower and upper analog buses together (64 channel).
- 0996: Connects analog bus Guard (G) to the LO line on the upper 32 channels (banks 4 to 7).

Example: Connecting the Analog Bus

In this example, as shown in Figure 2-9, the HI and LO terminals of bank 0 channel 0 are closed and the LO terminal of bank 4 channel 0 is closed connecting them to their COM terminals. Control relays 0990 and 0991 are automatically set open when configured for three-wire mode.

To connect bank 0 (both terminals) and bank 4 (LO terminal to Guard) to the analog bus, control relays 0992 and 0996 must be closed. To connect bank 0 and bank 4 to the analog bus, execute:

```
CLOS (@10992,10996)      !Close control relay 0992 to connect bank
                          0-3 to the analog bus. Close control relay
                          0996 to connect bank 4-7 LO terminal to
                          the analog bus Guard terminal
```

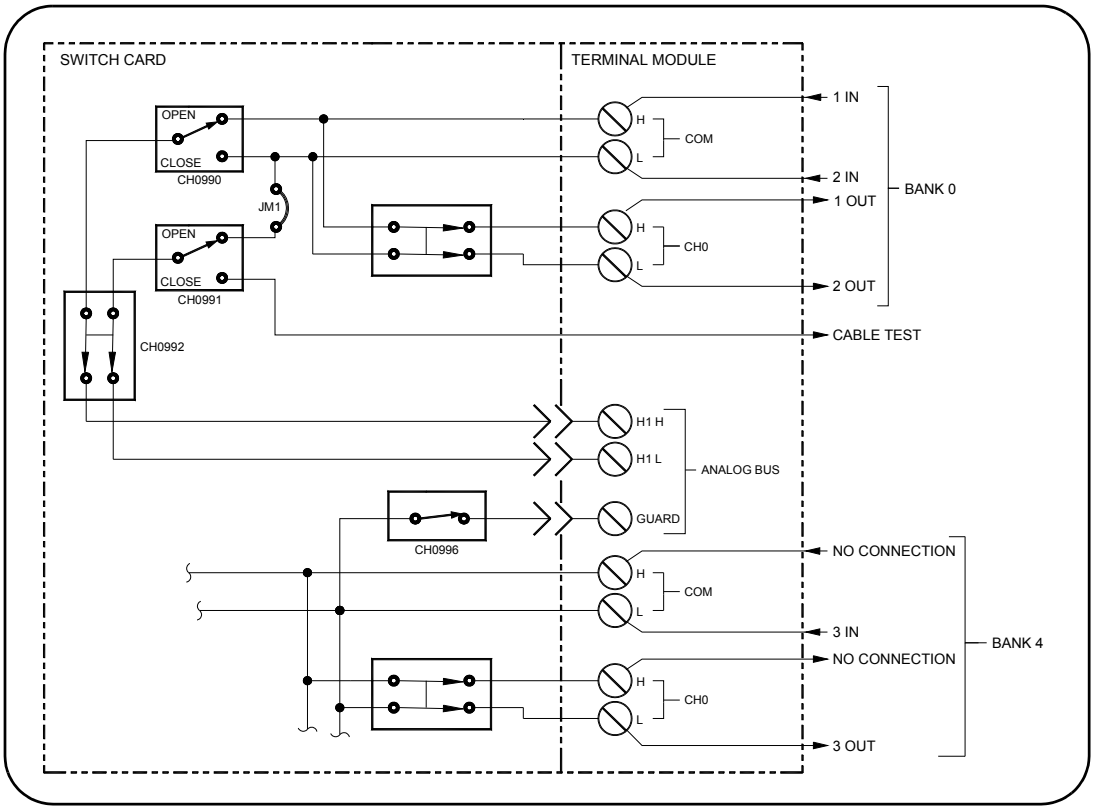


Figure 2-9. Example: Connecting the Analog Bus

Example: Cable Testing

You can connect and test multi-conductor cables or wiring harness conductors (for continuity) and insulators (for short circuit) using a single multiplexer module and system multimeter module.

Use the OPEN/CLOS <*channel_list*> commands to switch the control relays. For example, to close control relay 0992 (connect banks 0-3 to analog bus), execute CLOS (@10992).

The E1411B multimeter can directly measure channels of single or multiple multiplexer modules in a scanning multimeter configuration. The multimeter, when correctly programmed, automatically closes the appropriate control relays (0990-0996). For more information, see the *E1326B/E1411B User's Manual*.

The cable test example that follows can be expanded to test cables with more than 4 conductors. Continuity is checked by closing additional channel relays to test the odd numbered wires on the first loop pass, and the even numbered wires on the second loop pass (lines 60-180). Insulators are checked in groups of four wires by adding another loop (similar to lines 210 to 420).

For this example, the multiplexer module is configured to test a 4-conductor cable. Jumper JM1 must be removed to isolate the cable test terminal. Jumpers JM2 and JM3 must be removed to isolate bank 0 from bank 2.

Figure 2-10 shows how to connect the cable under test and the multimeter to the multiplexer. This example uses GPIB select code 7, primary address 09, and secondary address 03 for the multimeter and GPIB select code 7, primary address 09, and secondary address 14 for the multiplexer.

In the program, lines 10-30 set up multimeter for resistance measurement. Lines 40-50 reset multiplexer and switch banks 0-3 to analog bus. Lines 60-180 measure continuity of cable. The first loop pass measures wires 1 and 3 and the second loop pass measures wires 2 and 4.

Lines 190-420 measure insulation (short-circuit) of cable. The first loop pass measures wires 1 to 3, 2 to 3, 2 to 4, 1 to 4, and 1 to 2. The second loop pass measures wires 3 to 1, 4 to 1, 4 to 2, 3 to 2, and 3 to 4.

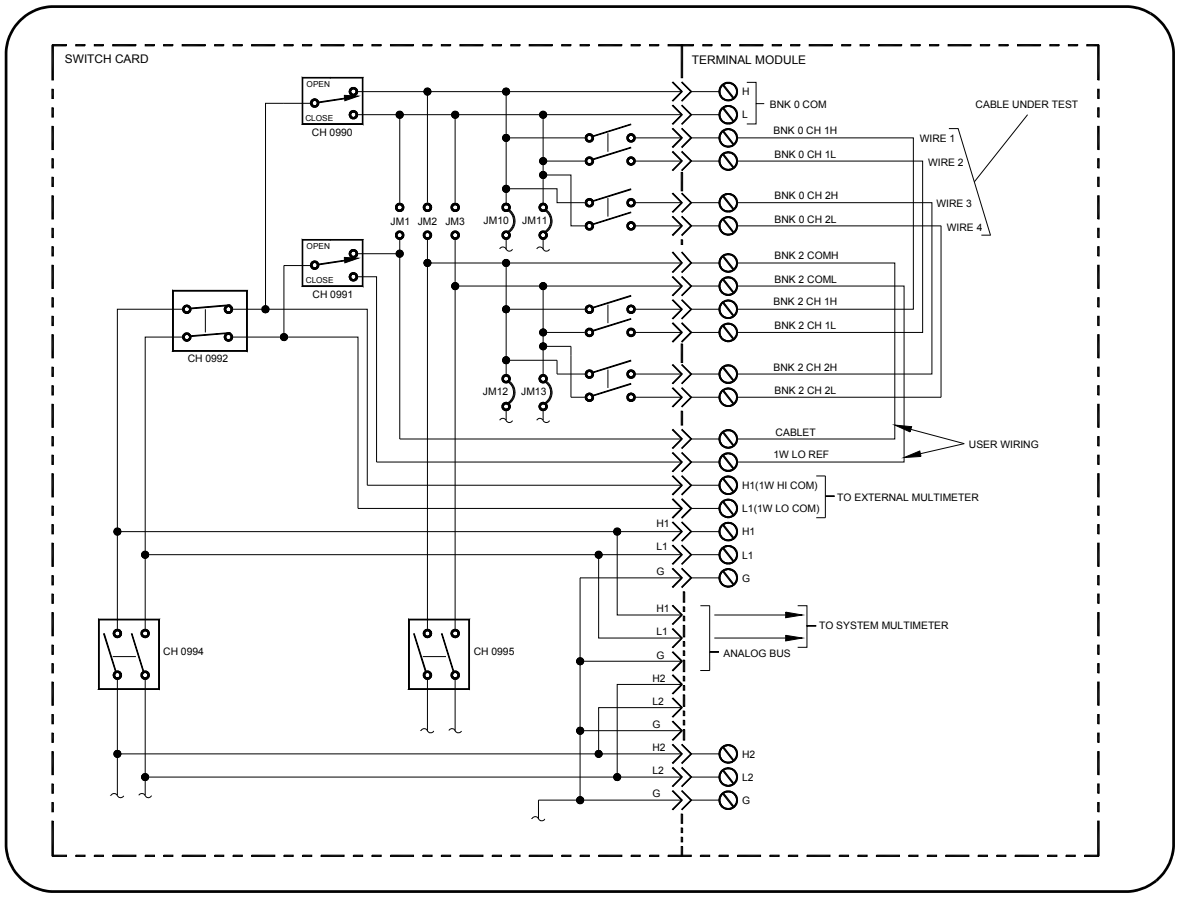


Figure 2-10. Example: Cable Testing

```
10    OUTPUT 70903;"*RST"
20    OUTPUT 70903;"CONF:RES;*OPC?"
30    ENTER 70903;Opc_
40    OUTPUT 70914;"*RST"
50    OUTPUT 70914;"CLOS (@10992)"
60    FOR I =1 TO 2
70       OUTPUT 70914;"CLOS (@101,121);*OPC?"
80       ENTER 70914;Opc_
90       OUTPUT 70903;"READ?"
100      ENTER 70903;A(I)
110      OUTPUT 70914;"OPEN (@101,121)"
120      OUTPUT 70914;"CLOS (@102,122);*OPC?"
130      ENTER 70914;Opc_
140      OUTPUT 70903;"READ?"
150      ENTER 70903;B(I)
160      OUTPUT 70914;"OPEN (@102,122)"
170      OUTPUT 70914;"CLOS (@10990,10991)"
180   NEXT I
190   OUTPUT 70914;"OPEN (@10990,10991)"
200   OUTPUT 70914;"CLOS (@122)"
210   FOR I = 1 TO 2
220      J = I + 100
230      OUTPUT 70914;"CLOS @";J;")"
240      OUTPUT 70903;"READ?"
250      ENTER 70903;C(I)
260      OUTPUT 70914;"CLOS (@10990)"
270      OUTPUT 70903;"READ?"
280      ENTER 70903;D(I)
290      OUTPUT 70914;"CLOS (@10991)"
300      OUTPUT 70903;"READ?"
310      ENTER 70903;E(I)
320      OUTPUT 70914;"OPEN (@10990)"
330      OUTPUT 70903;"READ?"
340      ENTER 70903;F(I)
350      OUTPUT 70914;"OPEN (@121,122)"
360      K = I + 120
370      OUTPUT 70914;"CLOS (@";K;")"
380      OUTPUT 70903;"READ?"
390      ENTER 70903;G(I)
400      OUTPUT 70914;"OPEN (@10990,10991)"
410      OUTPUT 70914;"OPEN (@":J;")"
420   NEXT I
430   PRINT "Continuity Wire # 1/2/3/4=";A(1),A(2),B(1),B(2)
440   PRINT "Insulation Wire # 1 to 2/3/4=";G(1),C(1),F(1)
450   PRINT "Insulation Wire # 2 to 1/3/4=";G(1),D(1),E(1)
460   PRINT "Insulation Wire # 3 to 1/2/4=";C(2),F(2),G(2)
470   PRINT "Insulation Wire # 4 to 1/2/3=";D(2),E(2),G(2)
480   END
```

Saving and Recalling States

The *SAV <*numeric_state*> command saves the current instrument state. The state number (0-9) is specified in the *numeric_state* parameter. The following settings are saved:

- Channel Relay States (bank 0-7 relays open or closed)
- Control Relay States (bank 9 relays open or closed)
- ARM:COUNt
- TRIGger:SOURce
- OUTPut[:STATe]
- INITiate:CONTinuous
- [ROUTe:]SCAN:MODE
- [ROUTe:]SCAN:PORT

The *RCL <*numeric_state*> command recalls a previously saved state. Enter the number (0-9) in the *numeric_state* parameter of the desired saved state. If *SAV was not previously executed using the selected number, the multiplexer module will configure to the reset values (refer to table 2-2).

NOTE **RCL, *RST, and *SAV do not affect the operating mode as set by the FUNCtion command or status register switch positions. If using the FUNCtion command to set the operating mode, the *SAV/*RCL command must be executed **AFTER** the FUNCtion command.*

Detecting Error Conditions

There are two general approaches to error checking: polling and using interrupts. An example of each method follows.

Example: Error Checking Using Polling

The simplest, but most time consuming, method is to ask the instrument whether there are errors at every step of the switching process. This is called *polling* and is illustrated in the following example.

```
10    DIM Err_num$[256]
20    OUTPUT 70914;"CLOS (@101)"
30    OUTPUT 70914;"SYST:ERR?"
40    ENTER 70914;Err_num$
50    IF VAL (Err_num$) 0 THEN
60        PRINT "Error";Err_num$
70        STOP
80    END IF
90    ...(program continues)
```

Example: Error Checking Using Interrupts

The second approach to error checking involves the use of interrupts. The following program is a method of checking for errors using interrupts as you program the multiplexer. The program monitors the multiplexer's Standard Event Status Register for an error condition. See the *E1406A Command Module User's Manual* for detailed information on the Standard Event Status Registers.

If no errors occur, the multiplexer functions as programmed. If errors do occur, the multiplexer interrupts the computer and the error codes and messages are read from the error queue. This example uses GPIB select code 7, primary address 09, and secondary address 14 for the multiplexer.

```
10   ON INTR 7 CALL Errmsg                    !Call computer subprogram "Errmsg" if a
                                               multiplexer programming error occurs
20   ENABLE INTR 7:2                          !Enable the computer to respond to an
                                               interrupt from the multiplexer
30   OUTPUT 70914;"*SRE 32"                   !Unmask the event status bit in the
                                               multiplexer's Status Register
40   OUTPUT 70914;"*ESE 64"                   !Unmask the error conditions in multiplexer
                                               Standard Event Status Register
50   .                                        !Program multiplexer for desired application
60   .
70   .
80   END
90   SUB Errmsg                               !Error Message subprogram
100    DIM Message$[256]
120    CLEAR 70914                            !When an error occurs, clear the multiplexer
                                               to regain control
130    B = SPOLL (70914)                      !Execute a serial poll to clear the service
                                               request bit in the Status Register
140    REPEAT
150        OUTPUT 70914;"SYST:ERR?"           !Read all error messages in the multiplexer
                                               error queue
160        ENTER 70914:Code,Message$
170        PRINT Code,Message$
180    UNTIL Code = 0
190    OUTPUT 70914;"*CLS"                    !Clear all bits in the multiplexer's Standard
                                               Event Status Register
200    STOP
210  SUBEND
```

Synchronizing the Multiplexer

This section discusses synchronizing the multiplexer module to other instruments when making measurements.

Example: Synchronizing Instruments

This example shows one way to synchronize instruments by switching a signal to be measured by a multimeter. This program verifies that the switching is complete before the multimeter begins a measurement.

The example uses GPIB select code 7, primary address 09, and secondary address 03 for the multimeter and GPIB select code 7, primary address 09, and secondary address 14 for the multiplexer.

```
10  OUTPUT 70914;"CLOS (@101);*OPC?"      !Close bank 1, channel 1 and request
                                           confirmation that the channel is closed
20  ENTER 70914;Opc_value
30  OUTPUT 70914;"CLOS? (@101)"           !Read confirmation
40  ENTER 70914;A
50  OUTPUT 70903;"MEAS:VOLT:DC?"          !Channel is confirmed closed, so the
                                           measurement can be made
60  ENTER 70903;Meas_value
70  PRINT Meas_value
80  END
```

Chapter 3
Relay Multiplexer Command Reference

About This Chapter

This chapter describes Standard Commands for Programmable Instruments (SCPI) and IEEE 488.2 Common Commands for the E1460A Relay Multiplexer module. See the appropriate command module user's manual for additional information on SCPI and Common Commands. This chapter contains the following sections:

- Command Types. 59
- SCPI Command Reference 61
- IEEE 488.2 Common Commands Quick Reference. 95
- SCPI Commands Quick Reference 96

Command Types

Commands are separated into two types: IEEE 488.2 Common commands and SCPI commands.

Common Commands Format

The IEEE 488.2 standard defines the Common commands that perform functions like reset, self-test, status byte query, etc. Common commands are four or five characters in length, always begin with an asterisk (*), and may include one or more parameters. The command keyword is separated from the first parameter by a space character. Some examples of Common commands are:

*RST, *ESE <*mask*>, *STB?

SCPI Commands Format

SCPI commands perform functions like closing switches, making measurements, and querying instrument states or retrieving data. A subsystem command structure is a hierarchical structure that usually consists of a top-level (or root) command, one or more lower-level commands, and their parameters. The following example shows part of a typical subsystem:

```
[ROUTe:]
    CLOSe <channel_list>
    SCAN <channel_list>
        :MODE?
```

[ROUTe:] is the optional root command, CLOSe and SCAN are second-level commands with parameters, and :MODE? is a third-level command. [ROUTe:] is an implied command and is, therefore, optional.

Command Separator

A colon (:) always separates one command from the next lower-level command, such as [ROUTe:]SCAN:MODE? Colons separate the root command from the second-level command ([ROUTe:]SCAN) and the second level from the third level (SCAN:MODE?).

Abbreviated Commands

The command syntax shows most commands as a mixture of upper- and lowercase letters. The uppercase letters indicate the abbreviated spelling for the command. For shorter program lines, send the abbreviated form. For better program readability, you may send the entire command. The instrument will accept either the abbreviated form or the entire command.

For example, if the command syntax shows DIAGnostic, DIAG and DIAGNOSTIC are both acceptable forms. Other forms of DIAGnostic, such as DIAGN or DIAGNOS will generate an error. You may use upper- or lowercase letters. Therefore, DIAGNOSTIC, diagnostic, and DiAgNoStIc are all acceptable.

Implied Commands

Implied commands appear in square brackets ([ ]) in the command syntax. *The brackets are not part of the command and are not sent to the instrument.* Suppose you send a second-level command but do not send the preceding implied command. In this case, the instrument assumes you intended to use the implied command and it responds as if you had sent it. Examine the portion of the [ROUTe] subsystem shown below:

```
[ROUTe:]
    CLOSe? <channel_list>
```

The root command [ROUTe:] is an implied command. To make a query about a channel's present status, you can send either of the following command statements:

```
ROUT:CLOSe? <channel_list> or CLOSe? <channel_list>
```

Variable Commands Syntax

Some commands have what appears to be a variable syntax, such as OUTPut:ECLTrg*n* and OUTPut:TTLTrg*n*. In these commands, the *n* is replaced by a number. No space is left between the command and the number because the number is not a parameter. The number is part of the command syntax. In the case of OUTP:ECLTrg*n*, *n* can range from 0 to 1. In OUTP:TTLTrg*n*, *n* can range from 0 through 7.

Parameter Types

The following table contains explanations and examples of parameter types you may see in this chapter.

Type	Explanations and Examples
Boolean	Boolean parameters represent a single binary condition that is either true or false (ON, OFF, 1, 0). Any non-zero value is considered true.
Discrete	Discrete parameters selects from a finite number of values. These parameters use mnemonics to represent each valid setting. An example is TRIGger:SOURce <*source*>, where *source* can be BUS, EXTernal, HOLD, IMMediate, ECLTrg*n*, or TTLTrg*n*.
Numeric	Numeric parameters are commonly used decimal representations of numbers including optional signs, decimal points, and scientific notation (for example, 123, 123E2, -123, -1.23E2, .123, 1.23E-2, 1.23000E- 01). Special cases include MIN, MAX, DEFault, and INFinity.
Optional	Optional parameters are shown within square brackets ([]). The brackets are not part of the command and are not sent to the instrument. If you do not specify a value for an optional parameter, the instrument chooses a default value. For example, consider ARM:COUNt?[MIN|MAX]. If you send the command without specifying a parameter, the present ARM:COUNt value is returned. If you send the MIN parameter, the command returns the minimum count available. If you send the MAX parameter, the command returns the maximum count available. Place a space between the command and the parameter.

Linking Commands

Linking IEEE 488.2 Common Commands with SCPI Commands. Use a semicolon (;) between the commands. For example, *RST;*RCL 1 or CLOS (@101);*SAV 1.

Linking Multiple SCPI commands. Use both a semicolon (;) and a colon (:) between the commands, such as CLOS (@101);:CLOS? (@101). SCPI also allows several commands within the same subsystem to be linked with a semicolon and colon, such as ROUT:CLOS (@101);:ROUT:CLOS? (@101).

SCPI Commands Reference

This section describes the Standard Commands for Programmable Instruments (SCPI) for the Relay Multiplexer module. Commands are listed alphabetically by subsystem and within each subsystem.

ABORt

The ABORt command stops a scan in progress when the scan is enabled via the interface and the trigger source is TRIGger:SOURce BUS or TRIGger:SOURce HOLD.

Subsystem Syntax

ABORt

Comments

ABORt Actions: The ABORt command terminates a scan in progress by causing the switchbox to no longer wait for a trigger. When the ABORt command is executed, the last channel switched during the scan remains in the position.

Stopping Scan Enabled Via Interface: When a scan is enabled via an interface, an interface clear command (CLEAR 7) can be used to stop the scan. When the scan is enabled via the interface and TRIG:SOUR BUS or HOLD is set, you can use ABORt to stop the scan.

Restarting a Scan: Use the INITiate command to restart the scan.

Related Commands: ARM, INITiate:CONTinuous, [ROUTe:]SCAN, TRIGger

Example

Stopping a Scan with ABORt

This example stops a (continuous) two-wire scan in progress in a single-module switchbox.

```
TRIG:SOUR BUS          !Trigger command will be via backplane
                        (bus) interface (*TRG command
                        generates trigger)
INIT:CONT ON           !Set continuous scanning
SCAN (@100:107)        !Scan channels 0 to 7 in bank 0

INIT                   !Start scan, close channel 0
  .
  .
  .
ABOR                   !Abort scan in progress
```

ARM

The ARM subsystem selects the number of scanning cycles (1 to 32767) for each INITiate command.

Subsystem Syntax

```
ARM
  :COUNt <number> MIN I MAX
  :COUNt? [MIN | MAX]
```

ARM:COUNt

ARM:COUNt *<number>* MIN | MAX allows scanning cycles to occur a multiple of times (1 to 32,767) with one INITiate command when INITiate:CONTinuous OFF | 0 is set. MIN sets 1 cycle and MAX sets 32,767 cycles.

Parameters

Name	Type	Range of Values	Default
<*number*>	numeric	1 thru 32,767 I MIN I MAX	1

Comments

Number of Scans: Use only numeric values between 1 and 32767, MIN, or MAX for the number of scanning cycles.

Related Commands: ABORt, INITiate:IMMediate

***RST Condition:** ARM:COUNt 1

Example

Setting Ten Scanning Cycles

This example sets a multiplexer module for 10 scans of channels 0 through 7 in bank 1 in a single-module switchbox.

```
ARM:COUN 10          !10 scans per INIT command
SCAN (@110:117)      !Scan channels 0 to 7 in bank 1
INIT                 !Start scan, close channel 0
```

ARM:COUNt?

ARM:COUNt? [MIN I MAX] returns the current number of scanning cycles set by ARM:COUNt. The current number of scan cycles is returned when MIN or MAX is not specified. With MIN or MAX as a parameter, MIN returns 1 and MAX returns 32,767.

Parameters

Name	Type	Range of Values	Default
<*MIN* | *MAX*>	numeric	MIN = 1, MAX = 32,767	current cycles

Comments

Related Commands: INITiate[:IMMediate]

Example

Query Number of Scans

This example sets a multiplexer module for 10 scanning cycles and queries the number of scan cycles set. The ARM:COUN? command returns 10.

```
ARM:COUN 10          !Set 10 scans per INIT command
ARM:COUN?            !Query number of scans
```

INITiate

The INITiate command subsystem selects continuous scanning cycles and starts the scanning cycle.

Subsystem Syntax

```
INITiate
  :CONTinuous <mode>
  :CONTinuous?
  [:IMMediate]
```

INITiate:CONTinous

INITiate:CONTinuous <*mode*> enables or disables continuous scanning cycles.

Parameters

Name	Type	Range of Values	Default
<*mode*>	boolean	0 I 1 I OFF I ON	OFF I 0

Comments

Continuous Scanning Operation: Continuous scanning is enabled with the INITiate:CONTinuous ON or INITiate:CONTinuous 1 command. Sending the INITiate:IMMediate command closes the first channel in the channel list. Each trigger from the source specified by the TRIGger:SOURce command advances the scan through the channel list. A trigger at the end of the channel list closes the first channel in the channel list and the scan cycle repeats.

Non-Continuous Scanning Operation: Non-continuous scanning is enabled with the INITiate:CONTinuous OFF or INITiate:CONTinuous 0 command. Sending the INITiate:IMMediate command closes the first channel in the channel list. Each trigger from the source specified by the TRIGger:SOURce command advances the scan through the channel list. At the end of the scanning cycle, the last channel in the channel list is closed and the scanning cycle stops.

Stopping Continuous Scan: See the ABORt command.

Related Commands: ABORt, ARM:COUNt, TRIGger

***RST Condition:** INITiate:CONTinuous OFF | 0

Example

Enabling Continuous Scanning

This example enables continuous scanning of bank 3, channels 0 through 7 of a switchbox. Since TRIGger:SOURce IMMediate (default) is set, the example uses an interface clear command (CLEAR 7) to stop the scan.

```
INIT:CONT ON              !Enable continuous scanning
SCAN (@130:137)           !Scan channels 0 to 7 in bank 3
INIT                      !Start scan, close channel 0
.
CLEAR 7                   !Stop scan cycle
```

INITiate:CONTinuous?

INITiate:CONTinuous? queries the scanning state. With continuous scanning enabled, the command returns "1" (ON). With continuous scanning disabled, the command returns "0" (OFF).

Example

Query Continuous Scanning State

This example enables continuous scanning of a switchbox and queries the state. Since continuous scanning is enabled, INIT:CONT? returns "1".

```
INIT:CONT ON              !Enable continuous scanning
INIT:CONT?                !Query continuous scanning state
```

INITiate[:IMMediate]

INITiate[:IMMediate] starts the scanning process and closes the first channel in the channel list. Successive triggers from the source specified by the TRIGger:SOURce command advance the scan through the channel list.

Comments

Starting the Scanning Cycle: The INITiate:IMMediate command starts scanning by closing the first channel in the channel list. Each trigger received advances the scan to the next channel in the channel list. An invalid channel list definition causes an error (see [ROUTe:]SCAN).

Stopping Scanning Cycles: See the ABORt command.

Example Enabling a Single Scan

This example enables a single scan of channels 0 through 7 in bank 5 of a single-module switchbox. The trigger source to advance the scan is immediate (internal) triggering set with TRIGger:SOURce IMMediate (default).

```
SCAN (@150:157)          !Scan channels 0 to 7 in bank 5
INIT                     !Begin scan, close channel 0 (use
                          immediate triggering)
```

OUTPut

The OUTPut command subsystem selects the source of the output trigger generated when a channel is closed during a scan. The selected output can be enabled, disabled, and queried. The three available outputs are the ECLTrg, TTLTrg trigger buses as well as the command module's (E1406A) front panel "Trig Out" port.

Subsystem Syntax

```
OUTPut
    :ECLTrgn< (:ECLTrg0 or ECLTrg1)
        [:STATe] <mode>
        [:STATe]?
    [:EXTernal]
        [:STATe] <mode>
        [:STATe]?
    :TTLTrgn (:TTLTrg0 through :TTLTrg7)
        [:STATe] <mode>
        [:STATe]?
```

OUTPut:ECLTrg*n*[:STATe]

OUTPut:ECLTrg*n*[:STATe] <*mode*> selects and enables which ECL Trigger bus line (0 or 1) will output a trigger when a channel is closed during a scan. This is also used to disable a selected ECL Trigger bus line. "*n*" specifies the ECL Trigger bus line (0 or 1) and "*mode*" enables (ON or 1) or disables (OFF or 0) the specified ECLTrg bus line.

Parameters

Name	Type	Range of Values	Default
<*n*>	numeric	0 or 1	N/A
<*mode*>	boolean	0 I 1 I OFF I ON	OFF I 0

Comments

Enabling ECL Trigger Bus: When enabled, a pulse is output from the selected ECL Trigger bus line (0 or 1) after each channel is closed during a scan. If disabled, a pulse is not output. The output is a negative-going pulse.

ECL Trigger Bus Line Shared by Switchboxes: Only one switchbox configuration can use the selected trigger at a time. When enabled, the selected ECL Trigger bus line (0 or 1) is pulsed by the switchbox each time a scanned channel is closed. To disable the output for a specific switchbox, send the OUTPut:ECLTrg*n* OFF or 0 command for that switchbox.

One Output Selected at a Time: Only one output (ECLTrg0 or 1; TTLTrg0, 1, 2, 3, 4, 5, 6, or 7; or EXTernal) can be enabled at one time. Enabling a different output source will automatically disable the active output. For example, if TTLTrg1 is the active output, and TTLTrg4 is enabled, TTLTrg1 will become disabled and TTLTrg4 will become the active output.

Related Commands: [ROUTe:]SCAN, TRIGger:SOURce, OUTPut:ECLTrg*n*[:STATe]?

***RST Condition:** OUTPut:ECLTrg*n*[:STATe] OFF (disabled)

Example

Enabling ECL Trigger Bus Line 0

```
OUTP:ECLT0:STAT 1          !Enable ECL Trigger bus line 0 to output
                            pulse after each scanned channel is
                            closed
```

OUTPut:ECLTrg*n*[:STATe]?

OUTPut:ECLTrg*n*[:STATe]? queries the present state of the specified ECL Trigger bus line. The command returns "1" if the specified bus line is enabled or "0" if the specified bus line is disabled.

Example

Query ECL Trigger Bus Enable State

This example enables ECL Trigger bus line 0 and queries the enable state. The OUTPut:ECLTrg*n*? command returns "1" since the port is enabled.

```
OUTP:ECLT0:STAT 1          !Enable ECL Trigger bus line 0
OUTP:ECLT0?                !Query bus enable state
```

OUTPut[:EXTernal][:STATe]

OUTPut[:EXTernal][:STATe] <*mode*> enables or disables the "Trig Out" port on the E1406A Command Module to output a trigger when a channel is closed during a scan. ON I 1 enables the port and OFF | 0 disables the port.

Parameters

Name	Type	Range of Values	Default
<*mode*>	boolean	0 I 1 I OFF I ON	OFF I 0

Comments

Enabling “Trig Out” Port: When enabled, a pulse is output from the “Trig Out” port after each scanned switchbox channel is closed. If disabled, a pulse is not output from the port after channel closures. The output is a negative-going pulse.

“Trig Out” Port Shared by Switchboxes: Only one switchbox configuration can use the selected trigger at a time. When enabled, the “Trig Out” port is pulsed by the switchbox each time a scanned channel is closed. To disable the output for a specific switchbox, send the OUTP OFF or 0 command for that switchbox.

One Output Selected at a Time: Only one output (ECLTrg0 or 1; TTLTrg0, 1, 2, 3, 4, 5, 6, or 7; or EXTernal) can be enabled at one time. Enabling a different output source will automatically disable the active output. For example, if TTLTrg1 is the active output and TTLTrg4 is enabled, TTLTrg1 will become disabled and TTLTrg4 will become the active output.

Related Commands: [ROUTe:]SCAN, TRIGger:SOURce, OUTPut[:EXTernal][:STATe]?

***RST Condition:** OUTPut[:EXTernal][:STATe] OFF (disabled)

Example

Enabling "Trig Out" Port

```
OUTP:EXT 1          !Enable "Trig Out" port to output pulse
                     after each scanned channel is closed
```

OUTPut[:EXTernal][:STATe]?

OUTPut[:EXTernal][:STATe]? queries the present state of the "Trig Out" port. The command returns "1" if the port is enabled or "0" if disabled.

Example

Query "Trig Out" Port Enable State

This example enables the "Trig Out" port and queries the enable state. The OUTPut? command returns "1" since the port is enabled.

```
OUTP:EXT ON         !Enable “Trig Out” port
OUTP:EXT?           !Query port enable state
```

OUTPut:TTLTrg*n*[:STATe]

OUTPut:TTLTrg*n*[:STATe] <*mode*> selects and enables which TTL Trigger bus line (0 to 7) will output a trigger when a channel is closed during a scan. This is also used to disable a selected TTL Trigger bus line. "*n*" specifies the TTL Trigger bus line (0 to 7) and "*mode*" enables (ON or 1) or disables (OFF or 0) the specified TTL Trigger bus line.

Parameters

Name	Type	Range of Values	Default
<*n*>	numeric	0 or 7	N/A
<*mode*>	boolean	0 I 1 I OFF I ON	OFF I 0

Comments

Enabling TTL Trigger Bus: When enabled, a negative-going pulse is output from the selected TTL Trigger bus line (0 to 7) after each channel in the switchbox is closed during a scan. If disabled, a pulse is not output.

TTL Trigger Bus Line Shared by Switchboxes: Only one switchbox configuration can use the selected TTL Trigger at a time. When enabled, the selected TTL Trigger bus line (0 to 7) is pulsed by the switchbox each time a scanned channel is closed. To disable the output for a specific switchbox, send the OUTPut:TTLTrgn OFF or 0 command for that switchbox.

One Output Selected at a Time: Only one output (ECLTrg0 or 1; TTLTrg0, 1, 2, 3, 4, 5, 6>, or 7; or EXTernal) can be enabled at one time. Enabling a different output source will automatically disable the active output. For example, if TTLTrg1 is the active output and TTLTrg4 is enabled, TTLTrg1 will become disabled and TTLTrg4 will be the active output.

Related Commands: [ROUTe:]SCAN, TRIGger:SOURce, OUTPut:TTLTrg*n*[:STATe]?

***RST Condition:** OUTPut:TTLTrg*n*[:STATe] OFF (disabled)

Example

Enabling TTL Trigger Bus Line 7

```
OUTP:TTLT7:STAT 1          !Enable TTL Trigger bus line 7 to output
                            pulse after each scanned channel is
                            closed
```

OUTPut:TTLTrg*n*[:STATe]?

OUTPut:TTLTrg*n*[:STATe]? queries the present state of the specified TTL Trigger bus line. The command returns "1" if the specified TTLTrg bus line is enabled or "0" if disabled.

Example

Query TTL Trigger Bus Enable State

This example enables TTL Trigger bus line 7 and queries the enable state. The OUTPut:TTLTrg*n*? command returns "1" since the port is enabled.

```
OUTP:TTLT7:STAT 1          !Enable TTL Trigger bus line 7
OUTP:TTLT7?                !Query bus enable state
```

[ROUTe:]

The [ROUTe:] command subsystem controls switching and scanning operations for multiplexer modules in a switchbox.

NOTE *This command opens all previously closed relays. Therefore, it should be the first relay configuration command.*

Subsystem Syntax

```
[ROUTe:]
    CLOSe <channel_list>>
    CLOSe? <channel_list>
    FUNCtion card_number, <function>
    FUNCtion? <card_number>
    OPEN <channel_list>
    OPEN? <channel_list>
    SCAN <channel_list>
        :MODE <mode>
        :MODE?
        :PORT <port>
        :PORT?
```

[ROUTe:]CLOSe

[ROUTe:]CLOSe <*channel_list*> closes the multiplexer channels specified by *channel_list*. *Channel_list* has the form (@*ss0hbc*) where *ss* = card number (01-99), *0h* = one-wire mode only high/low switching (00 or 01), *b* = bank number (0-7), and *c* = channel number (0-7).

Parameters

Name	Type	Range of Values	Default
<*channel_list*>	numeric	ss[00]00 to ss0177 ss00 to ss77 ss00 to ss037 ss00 to ss037	1-wire 2-wire 3-wire 4-wire

Comments

One-Wire Mode (WIRE1): When closing a channel in one-wire mode, the HI or LO line must be selected using *channel_list*. 01 selects HI, and 00 selects LO. If one-wire mode is selected, and a four-digit channel number is used, the LO line is selected.

Two-Wire Mode (WIRE2/WIRE2X64): Switches the HI and LO terminals of a channel in banks 0 through 3 or banks 4 through 7 to that bank's HI COM and LO COM terminals.

- **WIRE2:** Configures the E1460A as two independent 2x32 multiplexers.
- **WIRE2X64:** Switches the HI and LO terminals of a channel in banks 0 through 7 to that bank's HI COM and LO COM terminals. A maximum of 64 two-wire channels can be switched. This mode is available via E1406A (Switchbox Rev. A06.00 or later). Prior to this revision, closing control relay 0995 in two-wire mode will change the card configuration to a single 64-channel two-wire multiplexer.

Three/Four-Wire Modes (WIRE3/WIRE4): When closing a channel in three-wire or four-wire modes, only the lower bank (0-3) is specified. The upper bank pair (4-7) will automatically close the specified channel. If an attempt is made to close the upper bank pair (4-7) channels, an error will be generated.

Closing Channels: To close:

- a single channel, use ROUT:CLOS *(@ssbc*) or (@*ss0hbc*)
- multiple channels, use ROUT:CLOS (@*ssbc,ssbc,*...) or (@*ss0hbc,ss0hbc,*...)
- sequential channels, use ROUT:CLOS (@*ssbc:ssbc*) or (@*ss0hbc:ss0hbc*)
- groups of sequential channels, use ROUT:CLOS (@*ssbc:ssbc, ssbc:ssbc*) or (@*ss0hbc:ss0hbc, ss0hbc:ss0hbc*)
- or any combination of the above

NOTE *Channel numbers can be in the channel_list in any random order. However, closure order for multiple channels with a single command is not guaranteed.*

Closing the Control Relays: The control relays (0990 to 0996) can be closed to perform special functions (for example, connecting channels to the analog bus). Channels must be changed after the multiplexer has been configured using the [ROUTe:]FUNCtion command. Close:

- **0990** to select the LO terminal for one-wire switching
- **0991** to connect one-wire LO REF terminal to the one-wire LO COM terminal
- **0992** to connect lower 32 channels (banks 0 to 3) to the analog bus
- **0993** to connect upper 32 channels (banks 4 to 7) to the analog bus
- **0994** to connect lower and upper analog buses together
- **0995** to connect lower and upper common buses together (64-channel two-wire operation)
- **0996** to connect analog bus Guard to the LO line on the upper 32 channels (banks 4 to 7)

***OPC? Command:** Using the *OPC? command after the CLOSe command in your programs will ensure that the channel CLOSe command has executed prior to performing the next function (measure, read, etc.). This programming practice is highly recommended.

Related Commands: [ROUTe:]OPEN, CLOSe?, SCAN

***RST Condition:** All multiplexer channels are open.

Example

Closing Multiplexer Channels

This example closes channel 0, bank 0, in card 01, and channel 7, bank 6, in card 02 of a two-module switchbox. Both modules are in two-wire mode.

```
CLOS (@100,267)        !100 closes channel 0, bank 0 of card #1
                        and 267 closes channel 7, bank 6 of
                        card #2
```

[ROUTe:]CLOSe?

[ROUTe:]CLOSe? <*channel_list*> returns the current state of the channel(s) queried. *Channel_list* has the form (@*ssbc*) or (@*ss0hbc*) (see [ROUTe:]CLOSe for definition). The command returns "1" if channel(s) are closed or returns "0" if channel(s) are open.

Comments

Query is Software Readback: The ROUTe:CLOSe? command returns the current software state of the channel(s) specified. It does not account for relay hardware failures. A maximum of 128 channels at a time can be queried for a multiple-module switchbox.

Three/Four-Wire Modes (WIRE3/WIRE4): When configured for three- or four-wire modes, the upper bank pair (4-7) channels cannot be queried. If an attempt is made to query the upper bank pair (4-7) channels, an error will be generated.

Example

Query Channel Closure

This example closes channel 0, bank 0, in card 01, and channel 7, bank 6, in card 02 of a two-module switchbox and queries channel closure. Since the channels are programmed to be closed "1,1" is returned.

```
CLOS (@100,267)        !100 closes channel 0, bank 0, card #1 and
                        267 closes channel 7, bank 6, card #2
CLOS? (@100,267)       !Query state of channel 0, bank 0, card #1
                        and channel 7, bank 6, card #2
```

[ROUTe:]FUNCtion

[ROUTe:]FUNCtion <*card_number*>, <*function*> selects the operating mode of the multiplexer channels. All channels on the card specified by *card_number* operate in the specified mode. [ROUTe:] is NOT optional when ROUT:FUNC is used with a scanning multimeter configuration.

Parameters

Name	Type	Range of Values	Default
<*card_number*>	numeric	01 to 99	N/A
<*function*>	discrete	WIRE 1 I WIRE 2 | WIRE 2X64 I WIRE 3 I WIRE 4	WIRE 2

Comments

ROUTe is Not Always Optional: If used with a scanning multimeter configuration, ROUTe:FUNCtion must be used.

Command Not Always Used: This command is not required if the status register switch is configured to the desired mode.

Using the FUNCtion Command: When using the FUNCtion command to reconfigure the multiplexer to a different operating mode than the status register switch is set to, the command must be sent *AFTER* the card is powered up.

One-Wire Mode (WIRE1): Switches either the HI or LO terminal of a channel in banks 0 through 7, to the one-wire HI COM or one-wire LO COM terminal. When closing a channel in one-wire mode, the HI or LO line must be selected using *channel_list*. Only one of the 128 one-wire channels can be switched at a time.

Two-Wire Mode (WIRE2): Switches both the HI and LO terminals of a channel in banks 0 through 3 or banks 4 through 7 to that bank's HI COM and LO COM terminals.

Two-Wire Mode (WIRE2X64): Switches the HI and LO terminals of a channel in banks 0 through 7 to that bank's HI COM and LO COM terminals. A maximum of 64 two-wire channels can be switched. This mode is available via E1406A (Switchbox Rev. A06.00 or later). Prior to this revision, closing control relay 0995 in two-wire mode will change the card configuration to a single 64-channel two-wire multiplexer.

Three-Wire Mode (WIRE3): Banks are paired 0/4, 1/5, 2/6, and 3/7. Switches both the HI and LO terminal of a channel in bank 0-3, to that bank's HI and LO COM terminals. Also switches the LO terminal of the channel in pair bank 4-7 to that bank's LO COM terminal.

When closing a channel in three-wire mode, only the lower bank (0-3) is specified and the upper bank pair (4-7) will automatically close. A maximum of 32 three-wire channels can be switched. Selecting an upper bank (4-7) channel causes an error.

NOTE *In three-wire mode, do not connect user wiring to the HI terminal in the upper bank pair (4-7). This terminal is switched during three-wire operation, and dependent on relay configurations, could be switched to the HI COM terminal.*

Four-Wire Mode (WIRE4): Banks are paired 0/4, 1/5, 2/6, and 3/7. Switches both the HI and LO terminal of a channel in bank 0-3, to that bank's HI COM and LO COM terminals. Also switches the HI and LO terminal of the channel in pair bank 4-7, to that bank's HI and LO COM terminals. When closing a channel in four-wire mode, only the lower bank (0-3) is specified, and the upper bank pair (4-7) will automatically close. A maximum of 32 four-wire channels can be switched. Selecting an upper bank (4-7) channel causes an error.

Related Commands: [ROUTe:]OPEN, [ROUTe:]CLOSe, [ROUTe:]SCAN

***RST:** *RST does not change the selected mode.

Example

Configuring Multiplexer Mode

This example configures card 01 of a single-module switchbox to four-wire mode.

```
FUNC 1,WIRE4          !Configures card #1 to four-wire mode
```

[ROUTe:]FUNCtion?

[ROUTe:]FUNCtion? <*card_number*> returns the current operating mode of the card(s) queried. See [ROUTe:]FUNCtion for *card_number* definition. The command returns "WIRE1" if in the one-wire mode, "WIRE2" if in the two-wire mode, "WIRE3" if in the three-wire mode, or "WIRE4" if in the four-wire mode.

Example

Query Operating Mode

This example sets card #1 in a single-module switchbox to one-wire mode and queries the operating state. Since the one-wire mode is selected, "WIRE1" is returned.

```
FUNC 1,WIRE1          !Configure card #1 to one-wire mode
FUNC? 1               !Query mode of card #1
```

[ROUTe:]OPEN

[ROUTe:]OPEN <*channel_list*> opens the multiplexer channels specified by *channel_list*. *Channel_list* has the form (@*ss0hbc*) where *ss* = card number (00-99), *0h* = one-wire mode only high/low switching (00 or 01), *b* = bank number (0-7), and *c* = channel number (0-7).

Parameters

Name	Type	Range of Values	Default
<*channel_list*>	numeric	ss[00]00 to ss0177 ss00 to ss77 ss00 to ss037 ss00 to ss037	1-wire 2-wire 3-wire 4-wire

Comments

One-Wire Mode (WIRE1): When opening a channel in one-wire mode, the HI or LO line must be selected using *channel_list.* 01 selects HI and 00 selects LO. If one-wire mode is selected, and a four-digit channel number is used, the LO line is selected.

Two-Wire Mode (WIRE2): Switches both the HI and LO terminals of a channel in banks 0 through 3 or banks 4 through 7 to that bank's HI COM and LO COM terminals

Two-Wire Mode (WIRE2X64): Switches the HI and LO terminals of a channel in banks 0 through 7 to that bank's HI COM and LO COM terminals. A maximum of 64 two-wire channels can be switched. This mode is available via E1406A (Switchbox Rev. A06.00 or later). Prior to this revision, closing control relay 0995 in two-wire mode will change the card configuration to a single 64-channel two-wire multiplexer.

Three/Four-Wire Modes (WIRE3/WIRE4): When opening a channel in three-wire or four-wire modes, only the lower bank (0-3) is specified. The upper bank pair (4-7) will automatically open the specified channel. If an attempt is made to open the upper bank pair (4-7) channels, an error will be generated.

Opening Channels: To open:

- a single channel, use ROUT:OPEN (@*ssbc*) or (@*ss0hbc*)
- multiple channels, use ROUT:OPEN (@*ssbc,ssbc*,...) or (@*ss0hbc,ss0hbc*,...)
- sequential channels, use ROUT:OPEN (@*ssbc:ssbc*) or (@*ss0hbc:ss0hbc*)
- groups of sequential channels, use ROUT:OPEN (@*ssbc:ssbc, ssbc:ssbc*) or (@*ss0hbc:ss0hbc, ss0hbc:ss0hbc*)
- or any combination of the above

NOTE *Channel numbers can be in the channel_list in any random order. Hovever, opening order for multiple channels with a single command is not guaranteed.*

Opening the Control Relays: The control relays (0990 to 0996) can be opened to perform special functions (for example, isolating channels from the analog bus). Channels must be changed after the multiplexer has been configured using the [ROUTe:]FUNCtion command. Open:

- **0990** to select the HI terminal for one-wire switching
- **0991** to connect Cable Test terminal to the one-wire LO COM terminal
- **0992** to disconnect lower 32 channels (banks 0 to 3) from the analog bus
- **0993** to disconnect upper 32 channels (banks 4 to 7) from the analog bus
- **0994** to disconnect lower and upper analog buses
- **0995** to disconnect lower and upper common buses (dual 32-channel two-wire operation)
- **0996** to disconnect analog bus Guard from the LO line on the upper 32 channels (banks 4 to 7)

***OPC? Command:** Using the *OPC? command after the OPEN command in your programs will ensure that the channel OPEN command has executed prior to performing the next function (measure, read, etc.). This programming practice is highly recommended.

Related Commands: ROUTe:]CLOSe, [ROUTe:]OPEN?, [ROUTe:]SCAN

***RST Condition:** All multiplexer channels are open.

Example

Opening Multiplexer Channels

This example opens channel 0, bank 0, in card #1 and channel 7, bank 6, in card #2 of a two-module switchbox. Both modules are in two-wire mode.

```
OPEN (@100,267)          !100 opens channel 0, bank 0 of card #1
                          and 267 opens channel 7, bank 6 of
                          card #2
```

[ROUTe:]OPEN?

[ROUTe:]OPEN? <*channel_list*> returns the current state of the channel(s) queried. *Channel_list* has the form (@*ssbc*) or (@*ss0hbc*) (see [ROUTe:]OPEN for definition). The command returns "1" if channel(s) are open or returns "0" if channel(s) are closed.

Comments

Query is Software Readback: The ROUTe:OPEN? command returns the current software state of the channel(s) specified. It does not account for relay hardware failures. A maximum of 128 channels at a time can be queried for a multiple-module switchbox.

Three/Four-Wire Modes (WIRE3/WIRE4): When configured for three- or four-wire modes, the upper bank pair (4-7) channels cannot be queried. If an attempt is made to query the upper bank pair (4-7) channels, an error will be generated.

Example

Query Channel Open State

This example opens channel 0, bank 0, in card #1 and channel 7, bank 6, in card #2 of a two-module switchbox and queries the channel open states. Since the channels are programmed to be opened "1,1" is returned.

OPEN (@100,267)	*!100 opens channel 0, bank 0, card #1 and 267 opens channel 7, bank 6, card #2*
OPEN? (@100,267)	*!Query state of channel 0, bank 0, card #1 and channel 7, bank 6, card #2*

[ROUTe:]SCAN

[ROUTe:]SCAN <*channel_list*> defines the channels to be scanned. *Channel_list* has the form (@*ss0hbc*) where *ss* = card number (00-99), *0h* = one-wire mode only high/low switching (00 or 01), *b* = bank number (0-7), and *c* = channel number (0-7).

Parameters

Name	Type	Range of Values	Default
<*channel_list*>	numeric	ss[00]00 to ss0177 ss00 to ss77 ss00 to ss037 ss00 to ss037	1-wire 2-wire 3-wire 4-wire

Comment

Defining Scan List: When ROUTe:SCAN is executed, the channel list is checked for valid card, terminal, bank, and channel numbers. An error is generated for an invalid channel list.

64 Channel Limit: Individual channel numbers are limited to 64 due to the maximum length of command in the current driver.

Scanning Channels: To scan:

- a single channel, use ROUT:SCAN (@*ssbc*) or (@*ss0hbc*)
- multiple channels, use ROUT:SCAN (@*ssbc,ssbc,*...) or (@*ss0hbc,ss0hbc*,...)
- sequential channels, use ROUT:SCAN (@*ssbc:ssbc*) or (@*ss0hbc:ss0hbc*)
- groups of sequential channels, use ROUT:SCAN (@*ssbc:ssbc, ssbc:ssbc*) or <(@*ss0hbc:ss0hbc*, *ss0hbc:ss0hbc*)
- or any combination of the above

NOTE *Channel numbers can be in the channel_list in any random order.*

Scanning Operation: When a valid channel list is defined, INITiate[:IMMediate] begins the scan and closes the first channel in the *channel_list*. Successive triggers from the source specified by TRIGger:SOURce advance the scan through the *channel list*.

Stopping Scan: See the ABORt command.

Related Commands: OUTPut, TRIGger

***RST Condition:** All channels open.

Example

Scanning Using External Devices

See "Scanning Channels" in Chapter 2 for examples of scanning programs using external instruments.

[ROUTe:]SCAN:MODE

[ROUTe:]SCAN:MODE <*mode*> sets the multiplexer channels defined by the [ROUTe:]SCAN <*channel_list*> command for none, volts, two-wire ohms, or four-wire ohms measurements.

Parameters

Name	Type	Range of Values	Default
<*mode*>	discrete	NONE I VOLT I RES I FRES	NONE

Comments

Order of Command Execution: The [ROUTe:]SCAN:MODE and [ROUTe:]FUNCtion commands must be executed before the [ROUTe:]SCAN <*channel_list*> command.

[**ROUTe:]SCAN:MODE versus [ROUTe:]FUNCtion:FRES**: Measurement mode is not supported when FUNCtion is set to WIRE1 (one-wire mode).

NONE and VOLT Mode: When selected, *channel_list* is setup for volts measurements. VOLT mode is also used when making two-wire ohms measurements using two-wire multimeters.

RES Mode: When selected, *channel_list* is setup for two-wire ohms measurements. Control relay 0994 is closed when SCAN:PORT ABUS is selected. When selected, the multimeter SENSE/SOURCE leads are used to make the measurement. When using the HI/LO leads on a multimeter to make the measurement, use the VOLT mode.

FRES Mode: When selected, *channel_list* is setup for four-wire ohms measurements. When using four-wire ohms measurement mode, only the lower bank (0-3) is specified with the [ROUTe:]SCAN <*channel_list*> command. The upper bank pair (4-7) will automatically select the specified channel. Selecting an upper bank (4-7) channel causes an error.

***RST Condition:** [ROUTe:]SCAN:MODE NONE

Example

Selecting Four-Wire Ohms Measurements

This example selects four-wire ohms measurement mode (FRES) on card #1 of a single-module switchbox.

```
FUNC 1,WIRE4                !Set mode to four-wire
TRIG:SOUR EXT               !Selects external trigger source
SCAN:MODE FRES              !Selects four-wire W scan mode
SCAN (@130:137)             !Scan channels 0 to 7 in bank 3 (paired
                             with channels 0 to 7 in bank 7)
INIT                        !Starts scanning cycle
```

[ROUTe:]SCAN:MODE?

[ROUTe:]SCAN:MODE? returns the current state of the scan mode.

Comments

Values Returned. The command returns NONE, VOLT, RES, or FRES if the scan mode is in the none, volts, two-wire ohms, or four-wire ohms measurement mode, respectively.

Example

Query the Scanning Mode

This example selects the four-wire ohms measurement mode (FRES) on card #1 of a single-module switchbox, then queries the measurement state. Because four-wire ohms mode is selected, the query command returns "FRES".

```
SCAN:MODE FRES          !Select four-wire ohms scanning mode
SCAN:MODE?              !Query the scanning mode
```

[ROUTe:]SCAN:PORT

[ROUTe:]SCAN:PORT *<port>* enables or disables the closing of the analog bus connection control relays 0992, 0993, and 0996 during scanning. SCAN:PORT ABUS closes the appropriate control relay for analog bus connections. The ROUTe:SCAN:PORT NONE command prevents closing the control relays.

Parameters

Name	Type	Range of Values	Default
<*port*>	discrete	ABUS | NONE	NONE

Comments

Order of Command Execution: Measurement modes are selected by the [ROUTe:]FUNCtion and [ROUTe:]SCAN:MODE commands. Then the [ROUTe:]SCAN:PORT command, followed by the [ROUTe:]SCAN <*channel_list*> command.

Analog Bus Connection: The SCAN:PORT ABUS command only connects/disconnects the analog bus during scans. To connect/disconnect the analog bus when not scanning channels, it is necessary to switch the appropriate control relays (0992, 0993, and 0996). See the [ROUTe:]CLOSe or [ROUTe:]OPEN for more information.

***RST Condition:** [ROUTe:]SCAN:PORT NONE

Example Selecting the Analog Bus Port

This example selects the four-wire ohms measurement mode (FRES) on card #1 of a single-module switchbox and then enables the analog bus connection. Control relays 0992/0993 close and 0990/0991/0994/0995 open.

```
FUNC 1,WIRE4              !Set mode to four-wire
TRIG:SOUR EXT             !Select external trigger source
SCAN:MODE FRES            !Select the four-wire ohms mode
SCAN:PORT ABUS            !Select the analog bus port
SCAN (@130:137)           !Scan channels 0-7 in bank 3 (paired
                           with channels 0-7, bank 7)
INIT                      !Start scanning cycle
```

[ROUTe:]SCAN:PORT?

[ROUTe:]SCAN:PORT? returns the current state of the analog bus port. The command returns NONE if the analog bus connection control relays are disabled or ABUS if the control relays are enabled.

Example Query the Scan Port

This example selects the analog bus port and then queries the state. Because the analog bus port is selected, the query command returns "ABUS".

```
SCAN:PORT ABUS            !Select the analog bus port
SCAN:PORT?                !Query the port selection
```

STATus

The STATus subsystem reports the bit values of the Operation Status Register. It also allows you to unmask the bits you want reported from the Standard Event Register and to read the summary bits from the Status Byte Register.

Subsystem Syntax

```
STATus
    :OPERation
        :CONDition?
        :ENABle <number>
        :ENABle?
        [:EVENt?]
    :PRESet
```

The STATus system contains four registers, two of which are under IEEE 488.2 control: the Standard Event Status Register (*ESE?) and the Status Byte Register (*STB?). The operational status bit (OPR), service request bit (RQS), standard event summary bit (ESB), message available bit (MAV) and questionable data bit (QUE) in the Status Byte Register (bits 7, 6, 5, 4 and 3 respectively) can be queried with the *STB? command.

Use the *ESE? command to query the "*unmask*" value for the Standard Event Status Register (the bits you want logically OR'd into the summary bit). The registers are queried using decimal weighted bit values. The decimal equivalents for bits 0 through 15 are included in Figure 3-1.

A numeric value of 256 executed in a STAT:OPER:ENABle <*number*> command allows only bit 8 to generate a summary bit. The decimal value for bit 8 is 256.

The decimal values are also used in the inverse manner to determine which bits are set from the total value returned by an EVENt or CONDition query. The multiplexer driver exploits only bit 8 of Operation Status Register. This bit is called the scan complete bit which is set whenever a scan operation completes. Since completion of a scan operation is an event in time, bit 8 will never appear set when STAT:OPER:COND? is queried. However, you can find bit 8 set with the STAT:OPER:EVEN? query command.

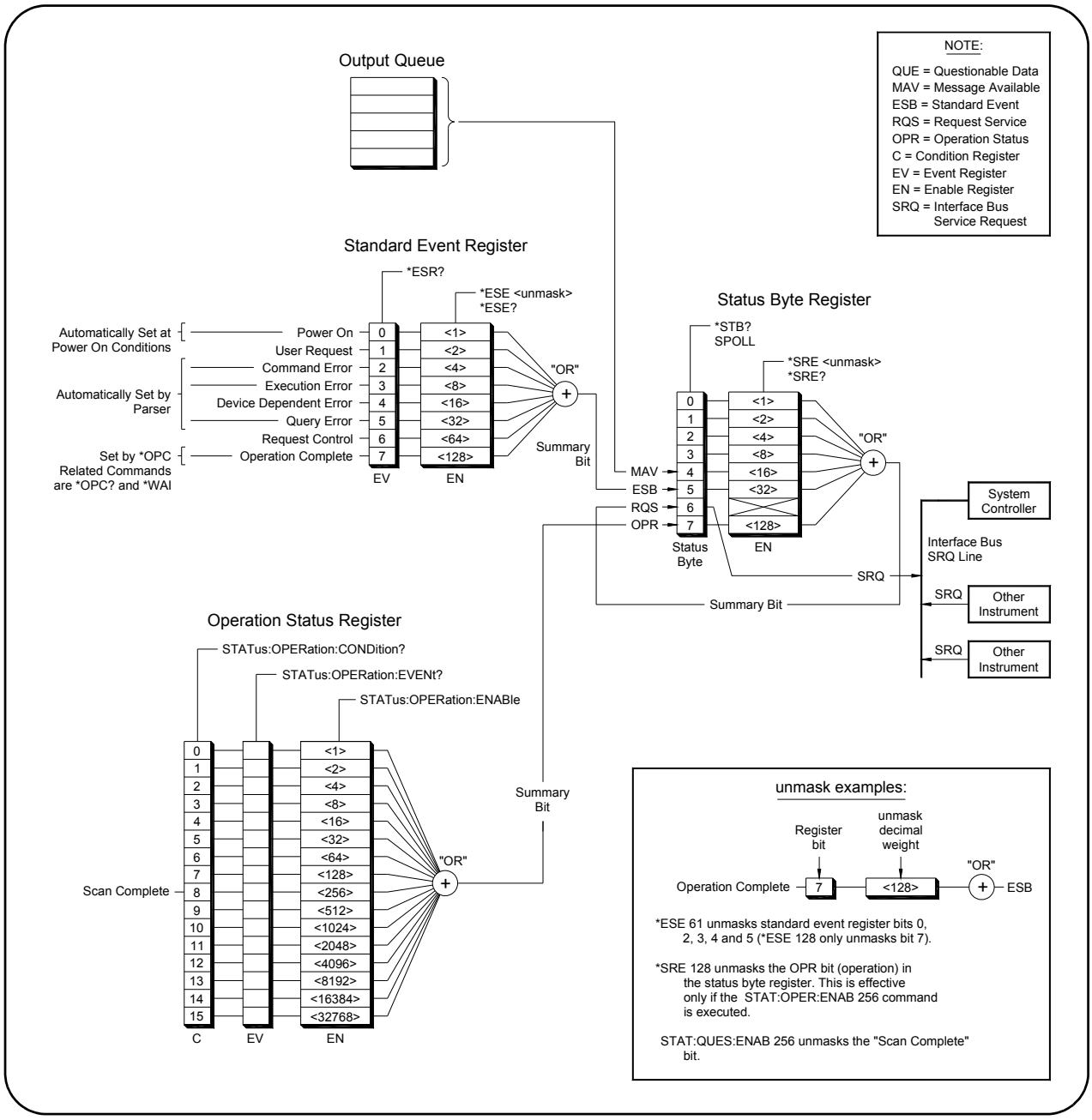


Figure 3-1. Relay Multiplexer Status System Registers

STATus:OPERation:CONDition?

STATus:OPERation:CONDition? returns the state of the Condition Register in the Operation Status Group. The state represents conditions which are part of the instrument's operation. The multiplexer driver does not set bit 8 in this register (see STATus:OPERation[:EVENt]?).

STATus:OPERation:ENABle

STATus:OPERation:ENABle *<number>* sets an enable mask to allow events recorded in the Event Register to send a summary bit to the Status Byte Register (bit 7). For multiplexer modules, when bit 8 in the Operation Status Register is set to 1 and that bit is enabled by the STATus:OPERation:ENABle command, bit 7 in the Status Register is set to 1.

Parameters

Name	Type	Range of Values	Default
<*number*>	numeric	1 through 65,535	N/A

Comments

Setting Bit 7 of the Status Register: STATus:OPERation:ENABle 256 sets bit 7 of the Status Register to 1 after bit 8 of the Operation Status Register is set to 1.

Related Commands: [ROUTe:]SCAN

Example

Enabling the Status Register

```
STAT:OPER:ENAB 256        !Enables bit 8 of the Operation Status
                           Register to be reported to bit 7 (OPR)
                           in the Status Register
```

STATus:OPERation:ENABle?

STATus:OPERation:ENABle? returns which bits in the Event Register (Operation Status Group) are unmasked.

STATus:OPERation[:EVENt]?

STATus:OPERation[:EVENt]? returns which bits in the Event Register (Operation Status Group) are set. The Event Register indicates when there has been a time-related instrument event.

Comments

Setting Bit 8 of the Operation Status Register: Bit 8 (scan complete) is set to 1 after a scanning cycle completes. Bit 8 returns to 0 (zero) after sending the STATus:OPERation[:EVENt]? command.

Returned Data after Sending STATus:OPERation[:EVENt]?: The command returns “+256” if bit 8 of the Operation Status Register is set to 1. The command returns “+0” if bit 8 of the Operation Status Register is set to 0.

Event Register Cleared: Reading the Event Register with the STATus:OPERation:EVENt? command clears it.

Aborting a scan: Aborting a scan will leave bit 8 set to 0.

Related Commands: [ROUTe:]SCAN

Example

Reading the Operation Status Register After a Scanning Cycle

STAT:OPER?	*!Returns the bit values of the Standard Operation Status Register*
read the register value	*!+256 shows bit 8 is set to 1 and +0 shows bit 8 is set to 0.*

STATus:PRESet

STATus:PRESet affects only the Enable Register by setting all Enable Register bits to 0. It does not affect either the "status byte" or the "standard event status". PRESet does not clear any of the Event Registers.

SYSTem

The SYSTem subsystem returns the numbers and messages in the error queue of a switchbox, and returns the switchbox module descriptions.

Subsystem Syntax

```
SYSTem
    :CDEScription? <number>
    :CPON <number> | ALL
    :CTYPe? <number>
    :ERRor?
```

SYSTem:CDEScription?

SYSTem:CDEScription? <*number*> returns the module description.

Parameters

Name	Type	Range of Values	Default
<*number*>	numeric	1 through 99	N/A

Comments

Multiplexer Module Description: The SYSTem:CDEScription? command returns the following E1460A descriptions, depending on mode currently configured:

- **One-Wire Mode:** "128 Channel S.E. Relay Mux"
- **Two-Wire Mode:** "Dual 32 Channel 2-Wire Relay Mux"
- **Two-Wire 64 Mode:** "64 Channel 2-Wire Relay Mux"
- **Three-Wire Mode:** "32 Channel 3-Wire Relay Mux"
- **Four-Wire Mode:** "32 Channel 4-Wire Relay Mux"

Example

Reading the Description of a Card #1 Module

This example selects the one-wire mode, then queries the description. Because one-wire mode is selected, the query command returns "WIRE1".

```
FUNC 1,WIRE1          !Set mode to one-wire
SYST:CDES?            !Return the description
```

SYSTem:CPON

SYSTem:CPON <*number*> | ALL sets the selected module (card) in a switchbox to its power-on state, with the exception of the mode selected.

Parameters

Name	Type	Range of Values	Default
<*number*>	numeric	1 through 99	N/A

Comments

Multiplexer Module Power-on State: The power-on state is all channels (relays) open. Note that SYSTem:CPON ALL and *RST opens all channels of all modules in a switchbox, while SYSTem:CPON <*number*> opens the channels in only the module (card) specified in the command. Current operating mode (as set by FUNCtion command) will not be affected by execution of the SYSTem:CPON <*number*> or *RST commands.

Example

Setting Card #1 Module to its Power-on State

```
SYST:CPON 1                    !Set card #1 to its power-on state
```

SYSTem:CTYPe?

SYSTem:CTYPe? <*number*> returns the module (card) type of a selected module in a switchbox.

Parameters

Name	Type	Range of Values	Default
<*number*>	numeric	1 through 99	N/A

Comments

64-Channel Multiplexer Module Model Number: The SYSTem:CTYPe? <*number*> command returns HEWLETT-PACKARD,El460A,0,A.02.00 where the 0 after E1460A is the module serial number (always 0) and A.02.00 is an example of the module revision code number.

Example

Reading the Model Number of a Card #1 Module

```
SYST:CTYP? 1                   !Returns the model number
```

SYSTem:ERRor?

SYSTem:ERRor? returns the error numbers and corresponding error messages in the error queue of a switchbox. See Appendix C for a listing of switchbox error numbers and messages.

Comments

Error Numbers/Messages in the Error Queue: Each error generated by a switchbox stores an error number and corresponding error message in the error queue. The error message can be up to 255 characters long.

Clearing the Error Queue: An error number/message is removed from the queue each time the SYSTem:ERRor? command is sent. The errors are cleared first-in, first-out. When the queue is empty, each following SYSTem:ERRor? command returns +0, “No error”. To clear all error numbers/messages in the queue, execute the *CLS command.

Maximum Error Numbers/Messages in the Error Queue: The queue holds a maximum of 30 error numbers/messages for each switchbox. If the queue overflows, the last error number/message in the queue is replaced by -350, “Too many errors”. The least recent error numbers/messages remain in the queue and the most recent are discarded.

***RST Condition:** *RST does not clear the error queue.

TRIGger

The TRIGger command subsystem controls the triggering operation of multiplexer modules in a switchbox.

Subsystem Syntax

```
TRIGger
    [:IMMediate]
    :SLOPe <slope>
    :SLOPe?
    :SOURce <source>
    :SOURce?
```

TRIGger[:IMMediate]

TRIGger[:IMMediate] causes a trigger event to occur when the defined trigger source is TRIGger:SOURce BUS or TRIGger:SOURce HOLD.

Comments

Executing the TRIGger[:IMMediate] Command: First, the measurement modes must be selected using the [ROUTe:]FUNCtion and [ROUTe:]SCAN:MODE commands. Then [ROUTe:]SCAN:PORT is selected, followed by the [ROUTe:]SCAN <*channel_list*> command and an INITiate[:IMMediate] command. All must be executed (unless defaults are used) before TRIGger[:IMMediate] will execute.

BUS or HOLD Source: If selected, the TRIGger:SOURce BUS or TRIGger:SOURce HOLD commands remain in effect after triggering a switchbox with the TRIGger[:IMMediate] command.

Related Commands: INITiate, [ROUTe:]SCAN

Example

Advancing Scan Using TRIGger Command

This example uses the TRIGger command to advance the scan of a single-module switchbox from bank 0, channels 0 through 7. Since TRIGger:SOURce HOLD is set, the scan is advanced one channel each time TRIGger is executed. For the example, ROUTe:SCAN:MODE and ROUTe:SCAN:PORT default values of NONE are used.

```
TRIG:SOUR HOLD          !Set trigger source to HOLD
SCAN (@100:107)         !Scan channels 0 to 7 in bank 0
INIT                    !Begin scan, close channel 00
loop statement          !Start count loop
TRIG                    !Advance scan to next channel
increment loop          !Increment loop count
```

TRIGger:SLOPe

TRIGger:SLOPe <*slope*> is used to select the polarity of the output trigger. For the E1460A, this command is not used.

Parameters

Name	Type	Range of Values	Default
<*slope*>	discrete	NEG	NEG

Comments

Command Not Supported. Attempting to change the TRIGger:SLOPe to anything other than NEG will generate an error.

TRIGger:SLOPe?

TRIGger:SLOPe? is used to query the polarity of the output trigger. For the E1460A, this query always returns NEG.

Example

Query Trigger Slope

```
TRIG:SLOP?                    !Always returns NEG
```

TRIGger:SOURce

TRIGger:SOURce <*source*> specifies the trigger *source* to advance the channel list during scanning.

Parameters

Name	Type	Range of Values	Default
BUS	discrete	*TRG or GET command	IMM
ECLTrg*n*	numeric	ECL Trigger bus line 0 or 1	IMM
EXTernal	discrete	"Trig In" port	IMM
HOLD	discrete	Hold Triggering	IMM
IMMediate	discrete	Immediate Triggering	IMM
Trg*n*	numeric	TTL Trigger bus line 0 - 7	IMM

Comments

Enabling the Trigger Source: The TRIGger:SOURce command only selects the trigger *source*. The INITiate[:IMMediate] command enables the trigger source.

Using the TRIGger Command: You can use TRIGger[:IMMediate] to advance the scan when TRIGger:SOURce BUS> or TRIGger:SOURce HOLD is selected.

One Trigger Input Selected at a Time: Only one input (ECLTrg0 or 1; TTLTrg0, 1, 2, 3, 4, 5, 6, or 7; or EXTernal) can be selected at one time. Enabling a different trigger source will automatically disable the active input. For example, if TTLTrg1 is the active input, and TTLTrg4 is enabled, TTLTrg1 will become disabled and TTLTrg4 will become the active input.

Using External Trigger Inputs: With TRIGger:SOURce EXTernal selected, only one switchbox at a time can use the external trigger input at the E1406A “Trig In” port. The trigger input is assigned to the first switchbox that requested the external trigger source (with an TRIGger:SOURce EXTernal command).

Using TTL or ECL Trigger Bus Inputs: With TRIGger:SOURce TTLTrg*n* or ECLTrg*n* selected, only one switchbox at a time can use the trigger bus selected on the E1406A Command Module bus. The trigger input is assigned to the first switchbox that requested the trigger source (with a TRIGger:SOURce TTLTrg*n* or ECLTrg*n* command). Only one of the ten available trigger bus lines (ECL0 to 1 or TTL0 to 7) can be specified at one time.

Assigning EXTernal I TTLTrg I ECLTrg Trigger Source: A switchbox assigned with TRIGger:SOURce EXT | TTLT | ECLT remains assigned to that source until the switchbox trigger source is changed to BUS, HOLD, or IMMediate. When the source is changed, the trigger source is available to the next switchbox that requests it (with a TRIGger:SOURce ECLT*n* command). If a switchbox requests a trigger already assigned to another switchbox, an error is generated.

Using Bus Triggers: To trigger the switchbox with TRIGger:SOURce BUS selected, use the IEEE 488.2 common command *TRG or the GPIB Group Execute Trigger (GET) command.

“Trig Out” Port Shared by Switchboxes: See the OUTPut command.

Related Commands: ABORt, [ROUTe:]SCAN, OUTPut

***RST Condition:** TRIGger:SOURce IMMediate

Example

Scanning Using External Triggers

This example uses external triggering (TRIG:SOUR EXT) to scan bank 0, channels 0 through 7 of a single-module switchbox. The trigger source to advance the scan is the input to the "Trig In" on an E1406A Command Module. When INIT is executed, the scan is started and bank 0, channel 0 is closed. Then, each trigger received at the "Trig In" port advances the scan to the next channel. For the example, ROUTe:SCAN:MODE and ROUTe:SCAN:PORT default values of NONE are used.

```
TRIG:SOUR EXT                 !Select external triggering
SCAN (@100:107)               !Scan channels 0 to 7 in bank 0
INIT                          !Begin scan, close bank 0, channel 0
trigger externally            !Advance scan to next channel
```

Example

Scanning Using Bus Triggers

This example uses bus triggering (TRIG:SOUR BUS) to scan bank 0, channels 0 through 7 of a single-module switchbox. The trigger source to advance the scan is the *TRG command (as set with TRIGger:SOURce BUS). When INIT is executed, the scan is started and bank 0, channel 0 is closed. Then, each *TRG command advances the scan to the next channel. For the example, ROUTe:SCAN:MODE and ROUTe:SCAN:PORT default values of NONE are used.

```
TRIG:SOUR BUS                 !Trigger command will be via backplane
                               (bus) interface (*TRG command
                               generates trigger)
SCAN (@100:107)               !Scan channels 0 to 7 in bank 0
INIT                          !Begin scan, close bank 0, channel 0
loop statement                !Loop to scan all channels
*TRG                          !Advance scan using bus triggering
increment loop                !Increment loop count
```

TRIGger:SOURce?

TRIGger:SOURce? returns the current trigger source for the switchbox. Command returns BUS, ECLT, EXT, HOLD, IMM, or TTLT for sources BUS, ECLTrg*n*, EXTernal, HOLD, IMMediate, or TTLTrg*n*, respectively.

Example

Querying the Trigger Source

This example sets external triggering and queries the trigger source. Since external triggering is set, TRIG:SOUR? returns "EXT".

```
TRIG:SOUR EXT                 !Set external trigger source
TRIG:SOUR?                    !Query trigger source
```

IEEE 488.2 Common Commands Reference

The following table lists the IEEE 488.2 Common (*) Commands that apply to the E1460A Relay Multiplexer module. For more information on Common Commands, see the applicable command module user's manual or the ANSI/IEEE Standard 488.2-1987.

Command	Title	Description
*CLS	Clear Status Register	Clears all Status Registers, the Request for OPC flag, and all Queues (except output queue).
*ESE *<mask>*	Event Status Enable	Sets the bits in the Standard Event Status Enable Register
*ESE?	Event Status Enable Query	Queries the current contents in the Standard Event Status Enable Register.
*ESR?	Event Status Register Query	Queries and clears current contents in the Standard Event Status Register.
*IDN?	Identification Query	Returns Identification String of the switchbox.
*OPC	Operation Complete	Sets the Request for OPC flag when all pending operations have completed. Also sets OPC bit in the Standard Event Status Register.
*OPC?	Operation Complete Query	Returns a "1" to the output queue when all pending operations have completed. Ensures synchronization between multiple instruments.
*RCL	Recall Saved State	Recalls previously stored multiplexer configuration. <*n*> (0 to 9) is location in memory where the desired (previously stored) set-up is located.
*RST	Reset	Opens all channels and invalidates current channel list for scanning. Sets ARM:COUN 1, TRIG:SOUR IMM, INIT:CONT OFF, OUTP:STAT OFF, SCAN:MODE NONE, and SCAN:PORT NONE
*SAV	Save Current State	Stores the current multiplexer configuration in memory. Stores current settings of the channel states. <*n*> (0 to 9) is the location in memory where the current set-up is to be stored.
*SRE *<mask>*	Service Request Enable	Sets the Service Request Enable Register bits and corresponding Serial Poll Status Register bits to generate a service request. Enable an event by specifying its decimal weight for *mask*.
*SRE?	Service Byte Enable Query	Queries current contents in the Service Request Enable Register.
*STB?	Status Byte Query	Queries the current contents in the Status Byte Register.
*TRG	Trigger	When scan is enable and trigger source to TRIG:SOUR BUS, use *TRG to trigger the switchbox to advance the scan.
*TST?	Self-Test Query (cc = card number with leading 0 deleted)	Returns +0 if self test passes. Returns +cc01 for firmware error. Returns +cc02 for bus error (communication problem with card). Returns +cc03 for bad ID information (ID Register on card). Returns +cc10 if an interrupt was expected but not received. Returns +cc11 if the busy bit was not held ~9 to 17 msec.
*WAI	Wait to Continue	Halts execution of commands and queries until No Operation Pending message is true.

SCPI Commands Quick Reference

The following table summarizes SCPI commands for the E1460A Relay Multiplexer module.

Command		Description
ABORt		Abort a scan in progress
ARM	:COUNt <*number*> MIN | MAX :COUNT? [MIN | MAX]	Multiple scans per INIT command Query number of scans
INITiate	:CONTinuous ON I OFF I 1 | 0 :CONTinuous? [:IMMediate]	Enables/Disables continuous scanning Query continuous scan state Starts a scanning cycle
OUTPut	:ECLTrg*n*[:STATe] ON | OFF 1| 0 :ECLTrg*n*[:STATe]? [:EXTernal][:STATe] ON I OFF 1 I 0 [:EXTernal][:STATe]? :TTLTrg*n*[:STATe]ON I OFF I 1 I 0 :TTLTrg*n*[:STATe]?	Enables/Disables ECL Trigger bus line pulse Query ECL Trigger bus line state Enables/Disables "Trig Out" pulse Query port enable state Enables/Disables TTL Trigger bus line pulse Query TTL Trigger bus line state
[ROUTe:]	CLOSe <*channel_list*> CLOSe? <*channel_list*> FUNCtion <*card_number*>,<*function*> FUNCtion? <*card_number*> OPEN <*channel_list*> OPEN? <*channel_list*> SCAN <*channel_list*> SCAN:MODE <*mode*> SCAN:MODE? SCAN:PORT <*port*> SCAN:PORT?	Close channel(s) Query channel(s) Set operating mode Query operating mode Open channel(s) Query channel(s) Define channels for scanning Set scan mode Query scan mode Select Analog Bus Query Analog Bus state
STATus	:OPERation:CONDition? :OPERation:ENABle <*number*> :OPERation:ENABle? :OPERation[:EVENt]? :PRESet	Returns status of Condition register Enables events in the Event register to be reported Returns which bits in the Event register are unmasked Returns which bits in the Event register are set Sets Enable register bits to 0
SYSTem	:CDEScription? <*number*> :CPON <*number*> I ALL :CTYPe? <*number*> :ERRor?	Returns description of module in switchbox Sets specified module in a switchbox to its power-on state Returns the module type Returns error number/message to error queue
TRIGger	[:IMMediate] :SLOPe <*slope*> :SLOPe? :SOURce BUS :SOURce ECLTrg*n* :SOURce EXTernal :SOURce HOLD :SOURce IMMediate :SOURce TTLTrg*n* :SOURce?	Causes a trigger to occur Select negative polarity of the output trigger Query polarity of the output trigger Trigger source is *TRG Trigger source is ECL Trigger bus line 0 or 1 Trigger source is "Trig In" port Hold off triggering Continuous (internal) triggering Trigger source is TTL Trigger bus line (0 - 7) Query current trigger source

Appendix A
Relay Multiplexer Specifications

Input Characteristics	
Maximum Voltage Terminal to Terminal: 220 Vdc; 250 Vac_{rms}	**Maximum Voltage Terminal to Chassis:** 220 Vdc; 250 Vac_{rms}
Maximum Current per Channel (non-inductive): 1 Adc or ac_{rms} (Vmax <30 Vdc or 250 V_{rms}) 0.3 Adc or ac_{rms} (Vmax <133 Vdc or 150 V_{rms})	**Maximum Power per Channel:** 40VA
Bias Current: From HI or LO to chassis, per group of 16 channels: <0.5 nA/Volt (at 25^0C, 25% RH)	
DC Performance	
Insulation Resistance (between any two points): >$5x10^6$ Ω at 40°C, 95% RH >$5x10^8$ Ω at 25°C, 40% RH	**Closed Channel Resistance:** <1.5 Ω initially <3.5 Ω at end of relay life
Maximum Thermal Offset per Channel: <7μV (differential H-L)	
AC Performance	
Minimum Bandwidth (-3dB, 50 W source/load): 2-Wire mode (4x16): >10 MHz 1-Wire mode (1x128): >3 MHz	**Crosstalk Between Channels @10 kHz:** 2-Wire mode (4x16): <-90 dB 1-Wire mode (1x128): <-60 dB
Open Channel Capacitance (channel to channel, channel to common): 2-Wire mode (4x16): <30 pF 1-Wire mode (1x128): <380 pF	**Closed Channel Capacitance (Hi-Lo, Lo-Chassis):** 2-Wire mode (4x16): 650/700 pF
General	
Module Size / Device Type: C-size VXIbus, Register based	**Power Requirements:** Voltage: +5 V +24 V Peak Module Current (A) 0.10 0.13 Dynamic Module Current (A) 0.10 0.02
Relay Life:[1] @ No Load: $5x10^6$ Operations @ Full (rated) Load: 10^5 Operations	**Watts/slot:** 5.0 **Cooling/slot:** 0.08 mm H_20 @ 0.42 Liter/sec
Terminals: Screw type, maximum wire size 16AWG	**Operating Temperature:** 0° - 55°C **Operating Humidity:** 65% RH, 0° - 40°C **Net Weight (kg):** 1.6

1 Relays are subject to normal wear-out based on the number of operations.

Notes:

Appendix B
Register-Based Programming

About This Appendix

This appendix contains information for register-based programming of the E1460A Relay Multiplexer module, including:

- Register Addressing 99
- Register Descriptions 102
- Programming Examples 107

Register Addressing

The E1460A Relay Multiplexer module is a register-based module that does not support the VXIbus word serial protocol. When a SCPI command is sent to the multiplexer, the E1406A Command Module parses the command and programs the multiplexer at the register level.

Register-based programming is a series of *reads* and *writes* directly to the multiplexer registers. This increases throughput speed since it eliminates command parsing and allows the use of an embedded controller. Also, if slot 0, the resource manager, and the computer (GPIB) interface are provided by other devices, a C-Size system can be downsized by removing the command module.

Register addresses for register-based devices are located in the upper 25% of VXI A16 address space. Every VXI device (up to 256 devices) is allocated a 32 word (64-byte) block of addresses. With twelve registers, the E1460A multiplexer uses twelve of the 64 addresses allocated.

Figure B-1 shows the register address location within A16 as it might be mapped by an embedded controller. Figure B-2 shows the location of A16 address space in the E1406A command module.

The Base Address

When you are reading or writing to a multiplexer register, a hexadecimal or decimal register address is specified. This address consists of a base address plus a register offset. The base address used in register-based programming depends on whether the A16 address space is outside or inside the command module.

A16 Address Space Outside the Command Module

When the E1406A command module is *not* part of your VXIbus system (see Figure B-1), the multiplexer's base address is computed as shown where "16" at the end of the address indicates a hexadecimal number.

$$C000_{16} + (LADDR * 64)_{16} \text{ or } 49{,}152 + (LADDR * 64)_{10}$$

where $C000_{16}$ (49,152) is the starting location of the register addresses, LADDR is the multiplexer's logical address, and 64 is the number of address logical address is 112 (70_{16}). If this address is not changed, the multiplexer will have a base address of:

$$C000_{16} + (112 * 64)_{16} = C000_{16} + 1C00_{16} = DC00_{16}$$

or

$$49{,}152 + (112 * 64) = 49{,}152 + 7168 = 56{,}320$$

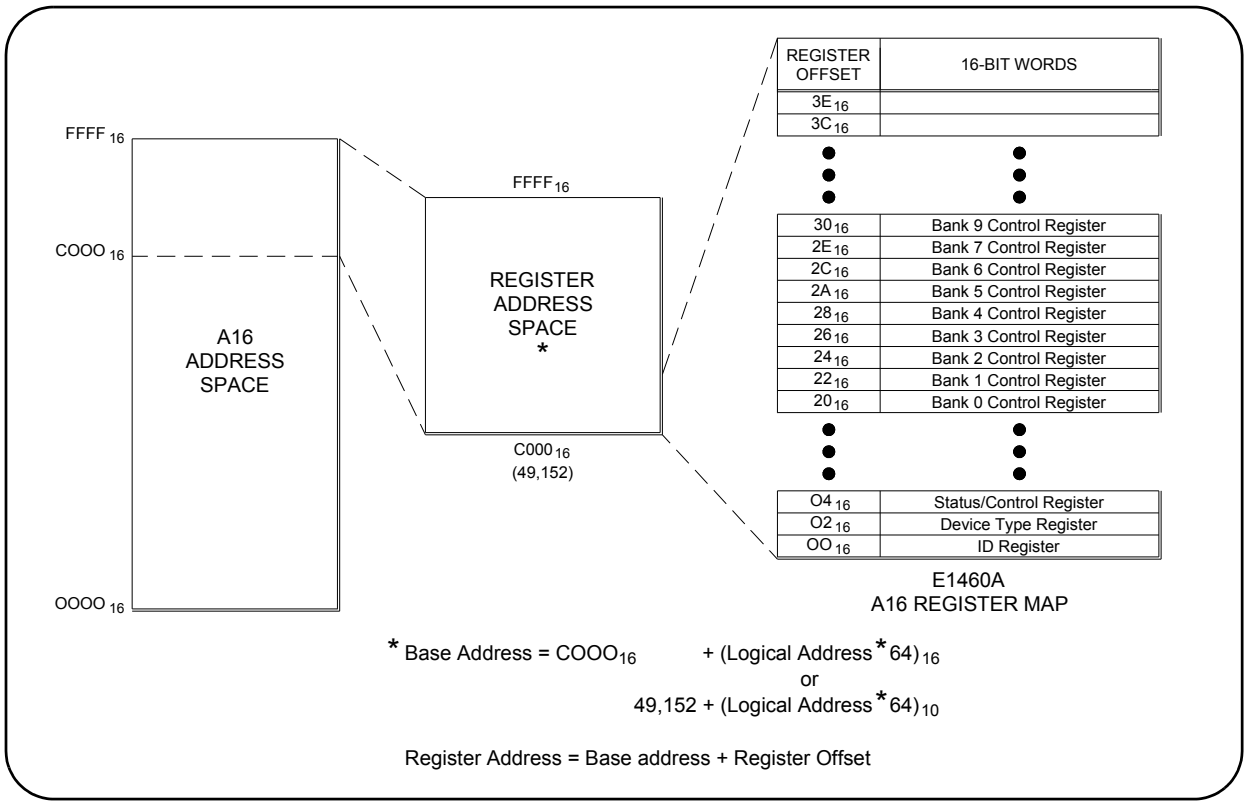


Figure B-1. Registers Within A16 Address Space (Outside the Command Module)

A16 Address Space Inside the Command Module or Mainframe

When the A16 address space is inside the E1406A Command Module (see Figure B-2), the multiplexer's base address is computed as:

$$1FC000_{16} + (LADDR * 64)_{16} \textit{ or } 2{,}080{,}768 + (LADDR * 64)_{10}$$

where $1FC000_{16}$ (2,080,768) is the starting location of the VXI A16 addresses, LADDR is the multiplexer's logical address, and 64 is the number of address bytes per register-based device. The multiplexer's factory set logical address is 112. If this address is not changed, the multiplexer will have a base address of:

$$1FC000_{16} + (112 * 64)_{16} = 1FC000_{16} + 1C00_{16} = 1FDC00_{16}$$

or

$$2{,}080{,}768 + (112 * 64) = 2{,}080{,}768 + 1536 = 2{,}087{,}936$$

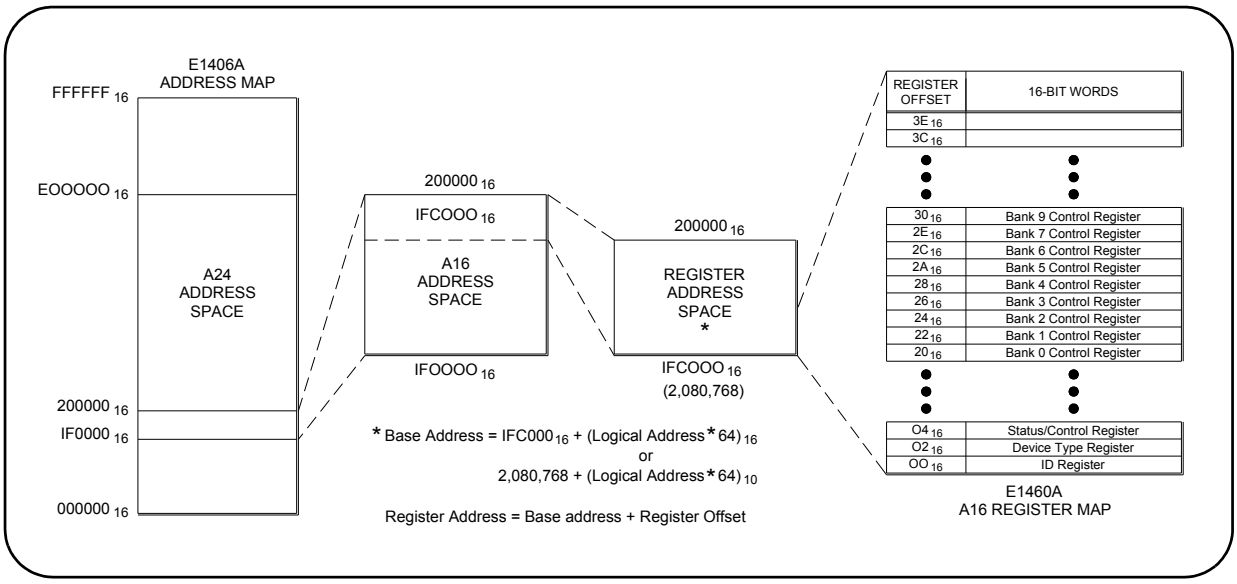


Figure B-2. Registers Within E1406 A16 Address Space

Register Offset

The *register offset* is the register's location in the block of 64 address bytes. For example, the multiplexer's Status/Control Register has an offset of 04_{16}. When you write a command to this register, the offset is added to the base address to form the register address:

$$DC00_{16} + 04_{16} = DC04_{16}$$

$$1FDC00_{16} + 04_{16} = 1FDC04_{16}$$

or

$$56{,}320 + 4 = 56{,}324$$

$$2{,}087{,}936 + 4 = 2{,}087{,}940$$

Register Descriptions

There are ten WRITE and twelve READ registers on the multiplexer. This section contains a description and a bit map for each register.

The WRITE Registers

You can write to the following multiplexer registers:

- Status/Control Register (base + 04_{16})
- Bank 0 Relay Control Register (base + 20_{16})
- Bank 1 Relay Control Register (base + 22_{16})
- Bank 2 Relay Control Register (base + 24_{16})
- Bank 3 Relay Control Register (base + 26_{16})
- Bank 4 Relay Control Register (base + 28_{16})
- Bank 5 Relay Control Register (base + $2A_{16}$)
- Bank 6 Relay Control Register (base + $2C_{16}$)
- Bank 7 Relay Control Register (base + $2E_{16}$)
- Channels 0990 - 0996 Relay Control Register (base + 30_{16})

The READ Registers

You can read the following multiplexer registers:

- Manufacturer ID Register (base + 00_{16})
- Device Type Register (base + 02_{16})
- Status/Control Register (base + 04_{16})
- Bank 0 Relay Control Register (base + 20_{16})
- Bank 1 Relay Control Register (base + 22_{16})
- Bank 2 Relay Control Register (base + 24_{16})
- Bank 3 Relay Control Register (base + 26_{16})
- Bank 4 Relay Control Register (base + 28_{16})
- Bank 5 Relay Control Register (base + $2A_{16}$)
- Bank 6 Relay Control Register (base + $2C_{16}$)
- Bank 7 Relay Control Register (base + $2E_{16}$)
- Channels 0990 - 0996 Relay Control Register (base + 30_{16})

Status/Control Register

You can perform reads and writes to the Status/Control Register (base + 04_{16}). The following table defines the Status/Control Register bits.

base + 04_{16}	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write*	Undefined									D	Undefined					
Read**	Undefined		S4	S3	S2	S1	Undefined		B	D	Undefined					

*D = Disable Interrupt by writing "1" in bit #6.
** B = Status "busy" is "0" in bit #7.
** D = Status "interrupt disable" is "1" in bit #6.
** S1-4 = Status "Configuration Status bits" as follows:
1-wire mode: bit #13 = 0, bit #12 = 0, bit #11 = 0, bit #10 = 1
2-wire dual 32 mode: bit #13 to 10 = all 0's or all 1's
2-wire 64 mode: bit #13 = 0, bit #12 = 0, bit #11 = 1, bit #10 = 0
3-wire mode: bit #13 = 0, bit #12 = 0, bit #11 = 1, bit #10 = 1
4-wire mode: bit #13 = 0, bit #12 = 1, bit #11 = 0, bit #10 = 0

Writing to the Status/Control Register

Writes to the Status/Control Register (base + 04_{16}) enable you to disable/enable the interrupt generated when channels are closed. To disable the interrupt generated when channels are closed, write a "1" to bit 6 of the Status/Control Register (base + 04_{16}). Typically, interrupts are only disabled to "peek-poke" a module. See the operating manual of your command module before disabling the interrupt.

Reading the Status/Control Register

Each relay requires about 12 msec execution time during which time the multiplexers are "busy". Bit 7 of this register is used to inform the user of a "busy" condition. The interrupt generated after a channel has been closed can be disabled. Bit 6 of this register is used to inform the user of the interrupt status.

For example, if the Status Register (base + 04_{16}) returns D3BF, the multiplexer module is not busy (bit 7 set), the interrupt is enabled (bit #6 = 0), and the configuration is four-wire (bits 10-13 set).

In addition, if a terminal module card is connected to the relay switch card, the present configuration of the terminal module card's status bit can be read. Bits 10, 11, 12, and 13 are used to determine the configuration of the terminal module card.

ID and Device Type Registers

ID Register: Reading this register returns: FFFF that shows that Hewlett-Packard as the manufacturer and that the module is an A16 register-based device.

base + 00_{16}	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Undefined															
Read	Manufacturer ID - returns $FFFF_{16}$ in Hewlett-Packard A16 only register-based card															

Device Type Register: Reading this register returns 0100_{16} if the device is the E1460A 64-Channel Multiplexer module.

base +02_{16}	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write	Undefined															
Read	0100_{16}															

Relay Control Registers

Writes to the Relay Control Registers (base + 20_{16} to 30_{16}) which enables you to switch the desired channel (banks 0-7) to COM or switch the desired Channel Relay Control Register (channels 0990 - 0996). Any number of relays per bank can be closed at a time.

Any bit pattern not indicated in the register maps result in the lowest-numbered channel being closed. For example, to connect both upper and lower banks to the analog bus, write a "1" to bits 2 and 3 of the (base + 30_{16}) to close bank 0990, relays 2 and 3. All other bits must be set to "0".

To reset the multiplexer (all relays open), you must write a "0" to each bit in the Relay Control Registers. Reading the Relay Control Registers always returns $FFFF_{16}$. Register maps for Bank 0 through Bank 7 Relay Control Registers and for the Channels 0990 - 0996 Relay Control Register follow. **r**

Bank 0 Relay Control Register

base +20_{16}	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write*	Undefined								CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Read	Always Returns FFFF16															

* Write a "1" to close channel to COM

Bank 1 Relay Control Register

base +22_{16}	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write*	Undefined								CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Read	Always Returns $FFFF_{16}$															

* Write a "1" to close channel to COM

Bank 2 Relay Control Register

base +24_{16}	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write*	Undefined								CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Read	Always Returns $FFFF_{16}$															

* Write a "1" to close channel to COM

Bank 3 Relay Control Register

base +26_{16}	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write*	Undefined								CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Read	Always Returns $FFFF_{16}$															

* Write a "1" to close channel to COM

Bank 4 Relay Control Register

base +28_{16}	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write*	Undefined								CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Read	Always Returns $FFFF_{16}$															

* Write a "1" to close channel to COM

Bank 5 Relay Control Register

base + $2A_{16}$	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write*	Undefined								CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Read	Always Returns $FFFF_{16}$															

* Write a "1" to close channel to COM

Bank 6 Relay Control Register

base + $2C_{16}$	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write*	Undefined								CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Read	Always Returns $FFFF_{16}$															

* Write a "1" to close channel to COM

Bank 7 Relay Control Register

base + $2E_{16}$	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write*	Undefined								CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Read	Always Returns $FFFF_{16}$															

* Write a "1" to close channel to COM

Channels 0990 - 0996 Relay Control Register

base + 30_{16}	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Write*	Undefined								CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Read	Always Returns $FFFF_{16}$															

* Write a "1" to close control relay

Programming Examples

Some examples follow to illustrate programming the multiplexer at the register level, including the following examples. Unless noted, each program is a C-language program.

- Example: Opening/Closing Multiplexer Channels
- Example: Using a Multimeter with the Multiplexer
- Example: Reading Module ID, Device Type, and Status Registers
- Example: Scanning Channels
- Example: Scanning Channels (HP-UX)

The C language programs were developed using Turbo C++ programming language on a PC connected via GPIB to the E1406A Command Module. "DIAG:POKE" and "DIAG:PEEK?" are the E1406A commands for direct register access.

NOTE *The examples in this section illustrate methods required when using a VXI slot 0 interface other than the E1406A, for which you would substitute the equivalent register access commands or functions. If a E1406A was used, you can use the E1460A SCPI driver in the E1406A firmware and register-based programming is not required.)*

Example: Opening/Closing Multiplexer Channels

The flowchart in Figure B-3 shows one way to close (or open) a multiplexer channel and determine when it has finished closing (or opening). The address of the multiplexer's Status Register is base + 04_{16}. The address of the channel is the base address plus the channel offset.

The multiplexer's Status Register bit 7 is monitored to determine when a multiplexer channel can be closed (or opened), and when a channel has finished closing (or opening). This C program example closes and then opend Channel 5 on bank 2 (register address 36). To initialize the E1460A, write zeros to all Relay Control bits.

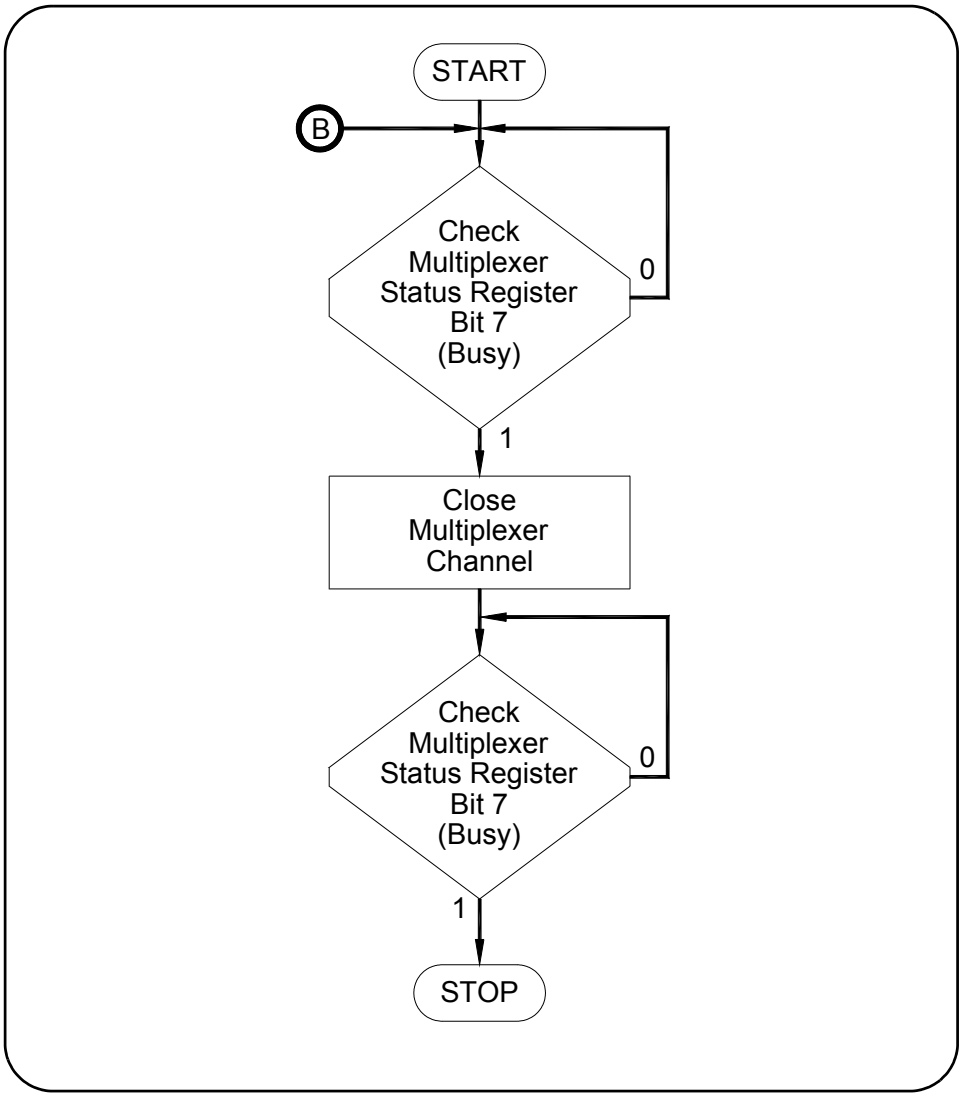


Figure B-3. Closing/Opening a Multiplexer Channel

```
#include <stdio.h>
#include <chpib.h>
#include <cfunc.h>

#define LOG_ADDR 112L
#define BASE_ADDR (long) ((0x1FC000) + (64 * LOG_ADDR))

main ()
{

        int reg_addr = 36;
        long bit_number = 5;
        float send_data[3], read;
        char state[2] = {13,10};

        send_data[1] = 16;
        send_data[2] = 1;

        send_data[0] = BASE_ADDR + reg_addr;

        IOEOI (7L, 0);IOEOL (7L, "", 0);
        IOOUTPUTS (70900L, "DIAG:POKE", 10);

        IOEOI (7L, 1);IOEOL (7L, state,0);
        IOOUTPUTA (70900L, send_data, 3);

        send_data[0] = BASE_ADDR + 4;

        IOEOI (7L, 0);IOEOL (7L," ", 0);
        IOUTPUTS (70900L, "DIAG:PEEK?", 11);
        IOEOI (7L, 1);IOEOL (7L, state, 2);
        IOOUTPUTA (70900L, send_data, 2);

        while (bit_number != 0)
        {

                IOENTER(70900L, &read);
                bit_number = ((long) (read)> 6 & 1);
        }
        return 0;

}
```

Example: Using a Multimeter with a Multiplexer

The flowchart in Figure B-4 shows a typical timing sequence between closing a multiplexer's channel and triggering a multimeter. The registers used are:

- Multiplexer Status Register (base + 04_{16})
- Multimeter Status Register (base + 04_{16})
- Multimeter Command Register (base + 08_{16})

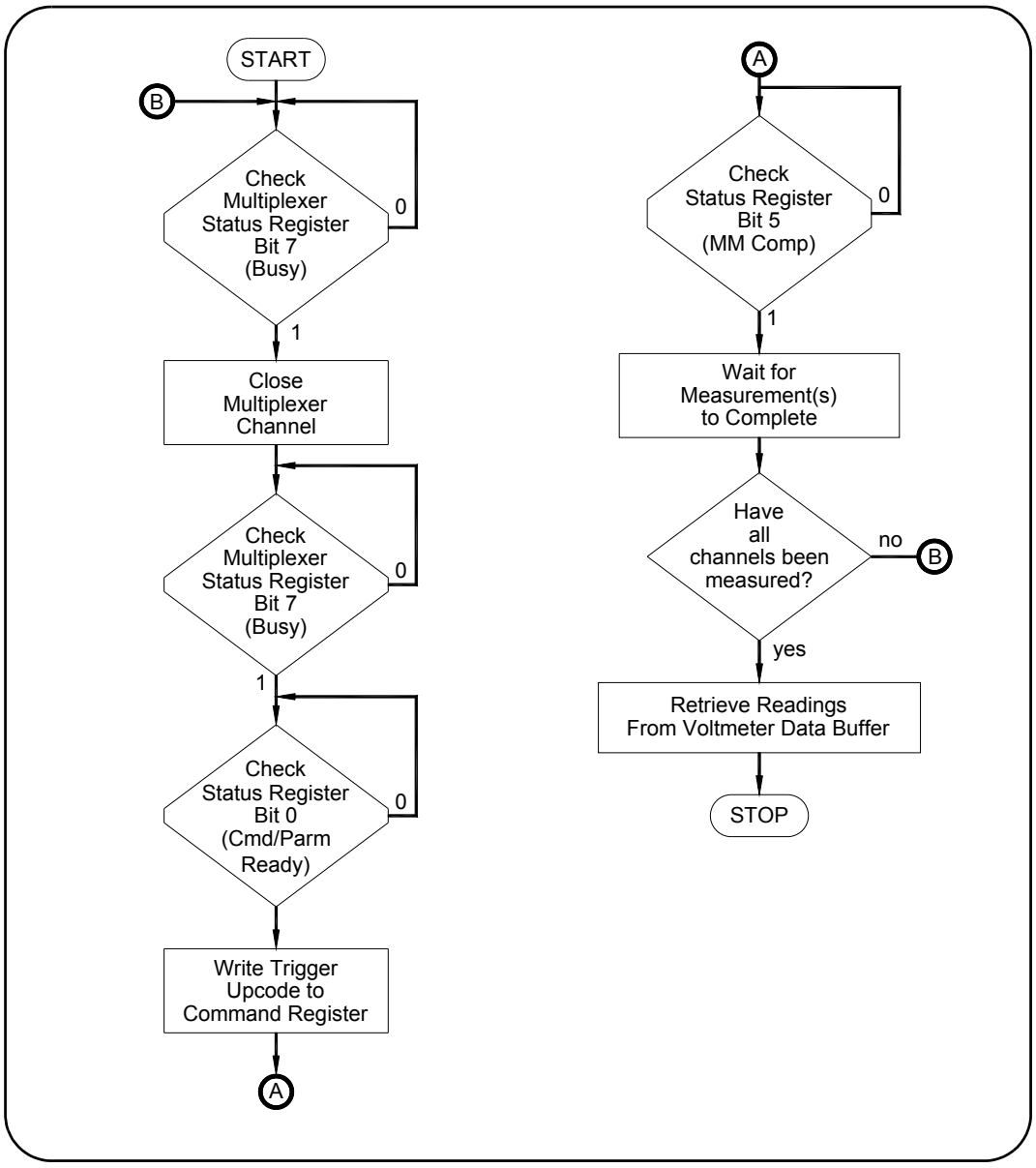


Figure B-4. Program Timing Between Multiplexer and Multimeter

The multiplexer's Status Register bit 7 is monitored to determine when a channel can be closed (or opened) and when a channel has finished closing (or opening). The ultimeter's Status bit 0 is monitored to determine when a trigger opcode can be written to the Command Register (the flowchart assumes the multimeter is already configured).

The multimeter's Status bit 5 is monitored to determine when the analog-to-digital (A/D) conversion is in progress and, thus, when to advance the channel. This enables each channel to be measured before the readings are read from the buffer. The multimeter's Autozero is often turned on to detect when bit 5 is active.

The channel can also be advanced by monitoring bit 4 (Data Ready). However, before measuring the next channel, readings from the previous channel must be read from the buffer in order to clear the bit.

Example: Reading ID, Device Type, and Status Registers

This C programming example reads the Module ID, Device Type, and Status Registers from the module.

```
#include <stdio.h>
#include <chpib.h>
#include <cfunc.h>
#define LOG_ADDR 112L
#define BASE_ADDR (long) ((0x1FC000) + (64 * LOG_ADDR))
main( )
{
        int  reg_addr;
        float send_data[3], read;
        char state[2] = {13,10};
        send_data[1] = 16;
        send_data[2] = 0;
        send_data[0] = BASE_ADDR + 0;
        IOEOI (7L, 0); IOEOL (7L, " ", 0);
        IOOUTPUTS (70900L, "DIAG:PEEK?", 11);
        IOEOI (7L, 1); IOEOL (7L, state, 2);
        IOOUTPUTA (70900L, send_data, 2);
        IOENTER(70900L, &read);
        printf("/nIdentification Register = %X",(int)read);
        send_data[0] = BASE_ADDR + 2;
        IOEOI (7L, 0);  IOEOL (7L, " " , 0);
        IOOUTPUTS (70900L, "DIAG:PEEK?", 11);
        IOEOI (7L, 1); IOEOL (7L, state, 2);
        IOOUTPUTA (70900L, send_data, 2);
        IOENTER(70900L, &read);
        printf("/nDevice Register = %X",(int)read);
        send_data[0] = BASE_ADDR + 4;
        IOEOI (7L, 0);       IOEOL (7L, "", 0);
        IOOUTPUTS (70900L, "DIAG:PEEK?" , 11);
        IOEOI (7L, 1);       IOEOL (7L, state, 2);
        IOOUTPUTA (70900L, send_data, 2);
        IOENTER(70900L, &read);
        printf("/nStatus Register = %X",(int)read);
        return 0;
}
```

Example: Scanning Channels

This C program example is similar to the closing/opening example except that it scans through the entire 64 channels on the multiplexer. By placing your own multimeter programming code where indicated you can create a scanning multimeter.

```
#include <stdio.h>
#include <chpib.h>
#include <cfunc.h>
#define LOG_ADDR 112L
#define BASE_ADDR (long) ((0x1FC000) + (64 * LOG_ADDR))
main ( )
{
        int reg_addr = 36;
        long bit_number;
        float send_data[3], read;
        char state[2] = {13,10};

        send_data[0] = BASE_ADDR + reg_addr;
        send_data[1] = 16;
        send_data[2] = pow(2,5);

        IOEOI (7L, 0); IOEOL (7L, " ", 0);
        IOOUTPUTS (70900L, "DIAG:POKE", 10);
        IOEOI (7L, 1); IOEOL (7L, state,0);
        IOOUTPUTA (70900L, send_data, 3);

        send_data[0] = BASE_ADDR + 4;
        IOEOI (7L, 0); IOEOL (7L, " ", 0);
        IOOUTPUTS (70900L, "DIAG:PEEK? ", 11);
        IOEOI (7L, 1); IOEOL (7L, state, 2);
        IOOUTPUTA (70900L, send_data, 2);
        while (bit_number != 0)
        {
            IOENTER(70900L, &read);
            bit_number = ((long) (read) >6 & 1); }

/* insert your multimeter programming code here*/

        send_data[2] = 0;
        IOEOI (7L, 0); IOEOL (7L, " ", 0);
        IOOUTPUTS (70900L, "DIAG:POKE", 10);
        IOEOI (7L, 1);IOEOL (7L, state,0);
        IOOUTPUTA (70900L, send_data, 3);

        send_data[0] = BASE_ADDR + 4;
        IOEOI (7L, 0); IOEOL (7L, " ", 0);
        IOOUTPUTS (70900L, "DIAG:PEEK?", 11);
        IOEOI (7L, 1); IOEOL (7L, state, 2);
        IOOUTPUTA (70900L, send_data, 2);

        while (bit_number != 0)
        {
```

```
            IOENTER(70900L, &read);
            bit_number = ((long) (read) >6 & 1);
        }
        return 0;
    }
```

Example: Scanning Channels (HP-UX) This example shows direct register programming using an E1499A (V/382) embedded computer running HP-UX and using the SICL interface library.

```
/*****************************************************************************
 *      Program to scan E1460A/68A/69A channels with a V/382       *
*****************************************************************************/

#include <stdio.h>
#include <fcntl.h>
#include <stdlib.h>
#include <sicl.h>
#include <time.h>

#define E1460A          "vxi,112"
/*Logical Address of device*/
#define BUSY              0x80
typedef unsigned short word;
typedef struct device_registers
 {
   word id_reg;
   word devtype_reg;
   word statcntl_reg;
   word dummy_reg[13];
   word bank0_reg;
   word bank1_reg;
   word bank2_reg;
   word bank3_reg;
   word bank4_reg;
   word bank5_reg;
   word bank6_reg;
   word bank7_reg;
   word bank99_reg;
}

DEVICE_REGISTERS;

main( )
 {

   INST          e1460a;

   int           i, j, id, rly;
   DEVICE_REGISTERS *dev_ptr;
   char          devstr[8];

   ionerror(I_ERROR_EXIT);
```

```
  /* Open a device session for the E1460A at laddr 112. */
  e1460a = iopen(E1460A);
  /* Map in the A16 registers  */
  dev_ptr=(DEVICE_REGISTERS *) imap(e1460a, I_MAP_VXIDEV, 0, 1, 0);
  /* Check card ID  */
  id=dev_ptr->>devtype_reg;
  if(id==0x0100)
  printf("Card identified as E1460A\n");

  else
{
  printf("Card not an E1460A - ID code: %hu\n",id);
}

  /* Open all relays */
  while(((dev_ptr->>statcntl_reg)&BUSY)==0);  dev_ptr->>bank0_reg=0x000;
  while(((dev_ptr->>statcntl_reg)&BUSY)==0);  dev_ptr->>bank1_reg=0x000;
  while(((dev_ptr->>statcntl_reg)&BUSY)==0);  dev_ptr->>bank2_reg=0x000;
  while(((dev_ptr->>statcntl_reg)&BUSY)==0);  dev_ptr->>bank3_reg=0x000;
  while(((dev_ptr->>statcntl_reg)&BUSY)==0);  dev_ptr->>bank4_reg=0x000;
  while(((dev_ptr->>statcntl_reg)&BUSY)==0);  dev_ptr->>bank5_reg=0x000;
  while(((dev_ptr->>statcntl_reg)&BUSY)==0);  dev_ptr->>bank6_reg=0x000;
  while(((dev_ptr->>statcntl_reg)&BUSY)==0);  dev_ptr->>bank7_reg=0x000;
  while(((dev_ptr->>statcntl_reg)&BUSY)==0);  dev_ptr->>bank99_reg=0x000;
  while(((dev_ptr->>statcntl_reg)&BUSY)==0);

  /* Close control relays 0992, 0993 & 0995 for 2X64 2-wire mode. */
  while(((dev_ptr->>statcntl_reg)&BUSY)==0);
  dev_ptr->>bank99_reg=0x02c;

  printf("Scanning bank 0, channels 0-7\n");
  rly = 1;
  for(i=0; iL8; i++)
{
  while(((dev_ptr->>statcntl_reg)&BUSY)==0);
  dev_ptr->>bank0_reg=rly;
     printf("Scanned bank 0 channel %d\n",i);
  rly = 2 * rly;
}
  printf("Done\n");
  exit(0);
}
```

Appendix C
Relay Multiplexer Error Messages

Table C-1 lists the error messages associated with the multiplexer module programmed with SCPI commands. See the appropriate command module user's manual for complete information on error messages.

Number	Title	Potential Causes
-211	Trigger Ignored	Trigger received when scan not enabled. Trigger received after scan complete. Trigger too fast.
-213	INIT Ignored	Attempting to execute an INIT command when a scan is already in progress.
-224	Illegal Parameter Value	Attempting to execute a command with a parameter not applicable to the command.
-350	Too Many Errors	The queue holds a maximum of 30 error numbers/messages for each switchbox. The queue has overflowed.
+1500	External Trigger Source Already Allocated	Assigning an external trigger source to a switchbox when the trigger source has already been assigned to another switchbox.
+2000	Invalid Card Number	Addressing a module (card) in a switchbox that is not part of the switchbox.
+2001	Invalid Channel Number	Attempting to address a channel of a module in a switchbox that is not supported by the module (for example).
+2006	Command Not Supported On This Card	Sending a command to a module (card) in a switchbox that is unsupported by the module.
+2008	Scan List Not Initialized	Executing a scan without the INIT command.
+2012	Invalid Channel Range	Invalid channel(s) specified in SCAN <*channel_list*> command. Attempting to begin scanning when no valid channel list is defined.
+2600	Function Not Supported On This Card	Sending a command to a module (card) in a switchbox that is not supported by the module or switchbox.
+2601	Channel List Required	Sending a command requiring a channel list without the channel list.

Notes:

Index
E1460A Relay Multiplexer User's Manual

A

A16 address space
- inside command module, 101
- outside command module, 100

ABORt subsystem, 62
addressing the multiplexer, 30
analog bus
- connecting, 22
- using, 52

ARM subsystem, 63
ARM:COUNt, 63
ARM:COUNt?, 64
attaching terminal modules, 28

B

base address, registers, 99

C

cautions, 15
channel addresses, 32
channel relay switches, 12
checking SCPI drivers, 29
command reference, 61
command types, 59
common commands
- *CLS, 95
- *ESE, 95
- *ESE?, 95
- *ESR?, 95
- *IDN?, 95
- *OPC, 95
- *OPC?, 95
- *RCL, 95
- *RST, 95
- *SAV, 95
- *SRE, 95
- *SRE?, 95
- *STB?, 95
- *TRG, 95
- *TST?, 95
- *WAI, 95
- command reference, 95
- format, 59

C (continued)

configuring terminal modules, 23
configuring the multiplexer, 15
configuring wire jumpers, 18
connecting the analog bus, 22
connecting user inputs, 25
control relays, 13

D

declaration of conformity, 9
detecting error conditions, 56
Device Type register, 104
documentation history, 8
downloading SCPI drivers, 30

E

error conditions, detecting, 56
error messages, 115
examples
- Advancing Scan Using TRIGger, 91
- Cable Testing, 53
- Closing Multiplexer Channels, 74
- Configuring Multiplexer Mode, 76
- Connecting the Analog Bus, 52
- Enabling "Trig Out" Port, 70
- Enabling a Single Scan, 67
- Enabling Continuous Scanning, 66
- Enabling ECL Trigger Bus Line 0, 69
- Enabling the Status Register, 86
- Enabling TTL Trigger Bus Line 7, 71
- Error Checking Using Interrupts, 57
- Error Checking Using Polling, 56
- Four-Wire Ohms Measurements, 81
- Initial Operation, 34
- Multiplexer Module Channel Lists, 33
- Opening Multiplexer Channels, 78
- Opening/Closing Multiplexer Channels, 108
- Querying "Trig Out" Port Enable State, 70
- Querying Channel Closure, 74
- Querying Channel Open State, 79
- Querying Continuous Scanning State, 66
- Querying ECL Trigger Bus Enable State, 69
- Querying Number of Scans, 64

E (continued)

examples (continued)
- Querying Operating Mode, 76
- Querying the Scan Port, 83
- Querying the Scanning Mode, 82
- Querying Trigger Slope, 92
- Querying TTL Trigger Bus Enable State, 71
- Querying the Trigger Source, 94
- Reading ID, Device Type, and Status Regs, 111
- Reading the Description of a Module, 88
- Reading the Operation Status Register, 87
- Scanning Channels, 112
- Scanning Channels (HP-UX), 113
- Scanning Channels Using 3457A Multimeter, 47
- Scanning Channels Using E1406A Cmd Mod, 44
- Scanning Channels Using E1412A Multimeter, 46
- Scanning Multimeter DCV Measurements, 49
- Scanning Multimeter Resistance Meas, 50
- Scanning Using Bus Triggers, 94
- Scanning Using External Devices, 80
- Scanning Using External Triggers, 93
- Selecting the Analog Bus Port, 83
- Setting Module to Power-on State, 89
- Setting Ten Scanning Cycles, 63
- Stopping a Scan with ABORt, 62
- Switching Channels (Four-Wire), 41
- Switching Channels (One-Wire), 38
- Switching Channels (Three-Wire), 40
- Switching Channels (Two-Wire), 39
- Synchronizing Instruments, 58
- Using a Multimeter with a Multiplexer, 110
- Using the Scan Complete Bit, 51

F

four-wire mode operation, 14

I

ID register, 104
IEEE 488.2 common commands reference, 95
initial operation, 34
INITiate subsystem, 65
INITiate:CONTinous, 65
INITiate:CONTinuous?, 66
INITiate[:IMMediate], 66
installing multiplexer in a mainframe, 21
interrupt priority, setting, 17

L

linking commands, 61
logical address switch, setting, 16

M

multiplexer
- addressing, 30
- analog bus, using, 52
- attaching terminal modules, 28
- block diagram, 12
- card numbers, 31
- channel addresses, 32
- channel relay switches, 12
- commands, 35
- components, 11
- configuration, 15
- connecting analog bus, 22
- connecting user inputs, 25
- control relays, 13
- description, 11
- detecting error conditions, 56
- error messages, 115
- four-wire mode, 14
- initial operation, 34
- installing in mainframe, 21
- one-wire mode operation, 14
- programming, 29
- query commands, 36
- register addressing, 99
- register offset, 101
- register-based programming, 99
- reset conditions, 36
- saving and recalling states, 56
- scan complete bit, 51
- scanning channels, 43
- switching channels, 37
- synchronizing, 58
- three-wire mode, 14
- two-wire mode operation, 14
- wiring terminal modules, 26

O

one-wire mode operation, 14
Option A3E, description, 23
OUTPut subsystem, 68
OUTPut:ECLTrgn[:STATe], 68
OUTPut:ECLTrgn[:STATe]?, 69
OUTPut[:EXTernal][:STATe], 69
OUTPut[:EXTernal][:STATe]?, 70
OUTPut:TTLTrgn[:STATe], 70
OUTPut:TTLTrgn[:STATe]?, 71

P

programming the multiplexer, 29

R

READ registers, description, 102
reconfiguring relay switch, 19
register addressing, 99
register offset, 101
register-based programming, 99, 107
registers
  Device Type, 104
  ID, 104
  Relay Control, 104
  Status/Control, 103
Relay Control registers, 104
relay multiplexer specifications, 97
relay switch, reconfiguring, 19
reset conditions, 36
restricted rights statement, 7
[ROUTe:] subsystem, 72
[ROUTe:]CLOSe, 72
[ROUTe:]CLOSe?, 74
[ROUTe:]FUNCtion, 75
[ROUTe:]FUNCtion?, 76
[ROUTe:]OPEN, 77
[ROUTe:]OPEN?, 79
[ROUTe:]SCAN, 79
[ROUTe:]SCAN:MODE, 80
[ROUTe:]SCAN:MODE?, 82
[ROUTe:]SCAN:PORT, 82
[ROUTe:]SCAN:PORT?, 83

S

safety symbols, 8
saving and recalling states, 56
scan complete bit, 51
scanning channels, 43
scanning multimeter, 11
SCPI commands
  abbreviated commands, 60
  command reference, 61
  command separator, 60
  format, 59
  implied commands, 60
  linking commands, 61
  parameter types, 61
  quick reference, 96
SCPI commands
  format, 30
  variable commands syntax, 60
SCPI drivers
  checking, 29
  downloading, 30
setting interrupt priority, 17
setting logical address switch, 16
setting status register switch, 17
specifications, 97
standard terminal module, description, 23
status register switch, setting, 17
STATus subsystem, 84
STATus:OPERation:CONDition?, 86
STATus:OPERation:ENABle, 86
STATus:OPERation:ENABle?, 86
STATus:OPERation[:EVENt]?, 87
STATus:PRESet, 87
Status/Control register, 103
switchbox, 11
switching channels, 37
synchronizing the multiplexer, 58
SYSTem subsystem, 88
SYSTem:CDEScription?, 88
SYSTem:CPON, 89
SYSTem:CTYPe?, 89
SYSTem:ERRor?, 90

T

terminal module Option A3E, description, 23
terminal modules
  attaching, 28
  configuring, 23
three-wire mode operation, 14
TRIGger[:IMMediate], 91
TRIGger:SLOPe, 92
TRIGger:SLOPe?, 92
TRIGger:SOURce, 92
TRIGger:SOURce?, 94
two-wire mode operation, 14

U

user inputs, connecting, 25

V

VXI Installation Consultant (VIC), 30

W

WARNINGS, 8
warnings, 15
warranty statement, 7
wire jumper functions, 19
wire jumpers, setting, 18
wiring terminal modules, 26
WRITE registers, description, 102

Notes: